Lecture Notes in Artificial Intelligence 2882

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Springer
Berlin
Heidelberg
New York
Hong Kong
London
Milan
Paris
Tokyo

Daniel Veit

Matchmaking in Electronic Markets

An Agent-Based Approach towards Matchmaking in Electronic Negotiations

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Author

Daniel Veit
University of Karlsruhe (TH)
Information Management and Systems
Englerstrasse 14
76131 Karlsruhe
Germany
E-mail: veit@iw.uka.de

Based on Dissertation 2003 DA 27, University Library at University of Karlsruhe

Title figure based on:
D.J. Veit, J.P. Müller, Ch. Weinhardt (2002)
Multidimensional Matchmaking for Electronic Markets
Journal of Applied Artificial Intelligence, 16(9-10), 2002, p. 858, Figure 1.

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at <http://dnb.ddb.de>.

CR Subject Classification (1998): I.2.11, I.2, K.4.4, J.1, H.3, H.4, C.2.4

ISSN 0302-9743
ISBN 3-540-20500-4 Springer-Verlag Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Boller Mediendesign
Printed on acid-free paper SPIN: 10967317 06/3142 5 4 3 2 1 0

Electronic negotiations concern the transaction of agreements on the b
electronic media, usually the Internet. They play an essential role wi
growing number of domains in the modern economy. Within recent
in consumer-to-consumer, business-to-consumer as well as in busin
business scenarios electronic platforms for all these fields have been
lished, and these support the participants in the information, the agre
or in the settlement phase of their transactions.

The degree to which the user is supported varies from platform to pla
At the low end, lists of hyperlinks at a Website provide support to th
Therefore, many authors define such platforms as electronic markets.
ever, this definition does not seem to be very reasonable, because und
definition all online catalogue systems form electronic markets. Since a
ket defines a location – even a virtual location – in which offers and re
meet and perform, several authors define an electronic market as a pla
in which at least all core phases – information, agreement and settlen
are electronically assisted.

A central component of electronic transactions is the matching of offeri
requesting parties within a market – the key issue in the above-men
agreement phase. Therefore, initially, knowledge has to be generated on
participants are potential transaction partners. There can be several
dates. For the final decision on which transaction partner to choose
essential to evaluate the alternatives. A so-called ranking of the altern
along with the preference of the participants is desirable.

Within his work, Mr. Veit is focusing exactly on this problem: How can i
ing offers be matched regarding a specific request. He defines a fram
in which a ranking can be generated in order to acquire an optimal
sion for a desired transaction. This ranking procedure within the fram
is called matchmaking. Hereby, the matchmaking procedure gets mor
more complex, particularly when not only flat structures are compare
multidimensional or multiattribute offers and requests meet, i.e., the
action decision depends on several decision criteria.

In this book Mr. Veit introduces a generic framework for multidimen
matchmaking, its implementation, and an analysis of it. The use cas

based on which matchmaking can be performed over arbitrary type
offer-request structures. The implementation – which has been realize
multi-agent system – can represent both offering and requesting agents
framework can be applied within a huge variety of application doma
electronic negotiations within electronic markets.

Therefore, this book is an essential contribution to the e-market/e-negot
literature and I wish the best success for its publication, for its autho
most of all for its readers.

September 2003

Christof Wein
University of California, Berkeley
Universität Karlsruhe, Ge

In the first place I would like to thank Prof. Christof Weinhardt fro
University of Karlsruhe who significantly contributed with his ideas
quality of this book. Also I would like to thank Prof. Rudi Studer,
Werner Rothengatter and Prof. Andreas Geyer-Schulz from the Univer
Karlsruhe for supporting my work.

I would like to especially say "thank you" to my wife Gudrun and m
little children Jonathan and Helena for the evenings which they spent w
me and thereby enabled me to finish this book.

Rudi Kober, Dr. Jörg P. Müller and Martin Schneider from Siemens
Munich also deserve special thanks for their support of my work.

For their inspirations, I would also like to thank Prof. Thomas Eite
the University of Technology, Vienna, Prof. V.S. Subrahmanian fro
University of Maryland, College Park, USA, and Prof. Bernhard Seege
the University of Marburg.

I would like to thank the human resources departments of Lufthansa
Frankfurt, the University Hospital in Giessen, and Siemens AG in M
for their contribution to the use cases in this book.

Last but not least I would like to thank all my colleagues in Karlsruhe fo
continuous support during the writing of this book. I would especial
to thank Dr. Miroslav Budimir, Clemens Czernohous, Michael Grune
Jörg Sandrock and Ilka Weber for proofreading the manuscript.

August 2003 Daniel
Universität Karlsruhe, Ge

Part I Fundamentals & Related Work

1 Introduction

Negotiation is a key component of e-commerce.
In automated negotiation, computational agents
find and prepare contracts on behalf of the
real-world parties they represent.
Tuomas Sandholm (1999)

The research domain of electronic negotiations – a new and interdisciplinary area – has become more and more important within the last years. Electronic negotiations are a fundamental mechanism to support automated decision-making among two or more involved parties by determining the best fitting counterpart for a mutually beneficial deal. Electronic negotiations and multidimensional matchmaking define important building blocks within the research area of electronic markets.

The topic of this work is *multidimensional matchmaking*, a process to determine quality rankings between offers and requests in electronic coordination scenarios. Especially in electronic negotiations matchmaking plays a central role. The general idea defining matchmaking methodologies is to provide automated mechanisms which allow a detailed and precise ranking on a set of offers to determine which offer fits a concrete request best (see Sycara et al. 1999).

At the same time, intelligent software agents start to evolve and gain importance in the research areas of automated negotiation. Since the discipline of intelligent software agents has become an integral component within the Distributed Artificial Intelligence (DAI) research, approaches which combine software agent technology with electronic negotiation capabilities from economic disciplines are presented (see Varian 1995).

Within the relationship between intelligent agents and electronic negotiations multi-attribute and – to the understanding of this work – *multidimensional* scenarios are of prime interest. Over the past few years these issues have become a central research aspect in economics and computer science research (see Bichler and Kalagnanam 2002; Sandholm 2000).

1.1 Motivation

The mission of this work is to contribute to the application, integration and impact of economic mechanisms and their design in modern, IT-based market infrastructures. A lot is to be done, applying fruitful research results from computer science areas to economic problems. This work contributes to this integration process which is gaining speed within the last years.

In electronic markets, services, goods and more generally said commitments are traded. In contrast to traditional markets, no human traders or human marketmakers are present to determine the best fitting counterparts for a deal. Instead, intelligent autonomous software individuals are authorized to enable market transactions on behalf of the human participants.

Many problems have to be faced when traditional markets are moving to electronic media. One central problem is multidimensional matchmaking among offer and request instances as a phase of complex multidimensional electronic negotiations. In markets offers and requests are representing the positions of the participants. Taking a look at different markets in different domains parallels can be drawn facing the matchmaking problem. In most cases, instances of texts, dates, prices and their domains have to be compared using intelligent mechanisms. After that, the ratings of these instances need to be combined.

The central research question in this work is the

- conception,
- definition,
- modelling,
- implementation,
- application and
- evaluation

of intelligent matchmaking mechanisms for electronic markets. The work provides a generic and domain independent modelling of the multidimensional matchmaking problem in electronic markets. This model is then applied to a specific market for which the generic matchmaking framework was adjusted. Several promising results have been derived.

In 1999 Siemens Business Services[1] and the Siemens AG Corporate Technology, Competence Center for Intelligent Autonomous Systems[2] initialized a project with the target customer Federal German Employment Agency[3] to develop an intelligent matchmaking component for electronic, web-based negotiations in the german human resources market.

[1] http://www.sbs.siemens.com (06/25/2002)
[2] http://www.ct.siemens.com (06/25/2002)
[3] http://www.arbeitsamt.de (06/25/2002)

This project background was ideally suited to develop a generic, configurable and intelligent matchmaking framework *Generic Request Architecture for Passive Provider Agents*, short GRAPPA , together with Siemens AG CT. GRAPPA has been patented internationally (China, Japan, USA and several countries in Europe) under publication number WO02/06974 (file number PCT/DE01/02407 and DE10034694.4) and applied in a large scale commercial prototype.

1.2 Reader's Guide

This work contributes a definition of matchmaking fundamentals, an overview on matchmaking research as well as a structured and formal definition of a generic matchmaking framework. An implementation of this framework is introduced as well as a real-world application of the framework to a real life market. This market application is evaluated in comparison to human resources specialists' opinions as well as state-of-the-art indexing methods.

1.2.1 Structure of This Work

Figure 1.1 illustrates the structure of this work. Part I comprises the introduction in this chapter as well as the overview, definition and framework in Chapter 2. Chapter 3 presents related work. The reader may focus on Chapter 2 which provides next to terminology, general understanding and key definitions.

Part II provides the multidimensional matchmaking model and implementation. In Chapter 4 a formal model for the GRAPPA matchmaking system is introduced. The implementation issue is described in Chapter 5. For implementation details on the framework the reader is referred to the `javadoc` documentation of the GRAPPA framework.

Finally, Part III contains the real-world application of GRAPPA in the human resources domain – the HRNETAGENT project. Here, human resources information systems (HRIS) and HR-based enterprise resource planning (ERP) systems are introduced in Chapter 6. Chapter 7 contains the empirical evaluation of the GRAPPA framework in three branches:

- Siemens AG Corporate Technology HR, Munich, Germany.
- Lufthansa Cargo AG HR, Frankfurt, Germany.
- University Hospital Giessen Internal Medicine HR, Giessen, Germany.

The results of the evaluations are presented in Chapter 7. Finally this work is reviewed and an outlook on the future work is given in Chapter 8.

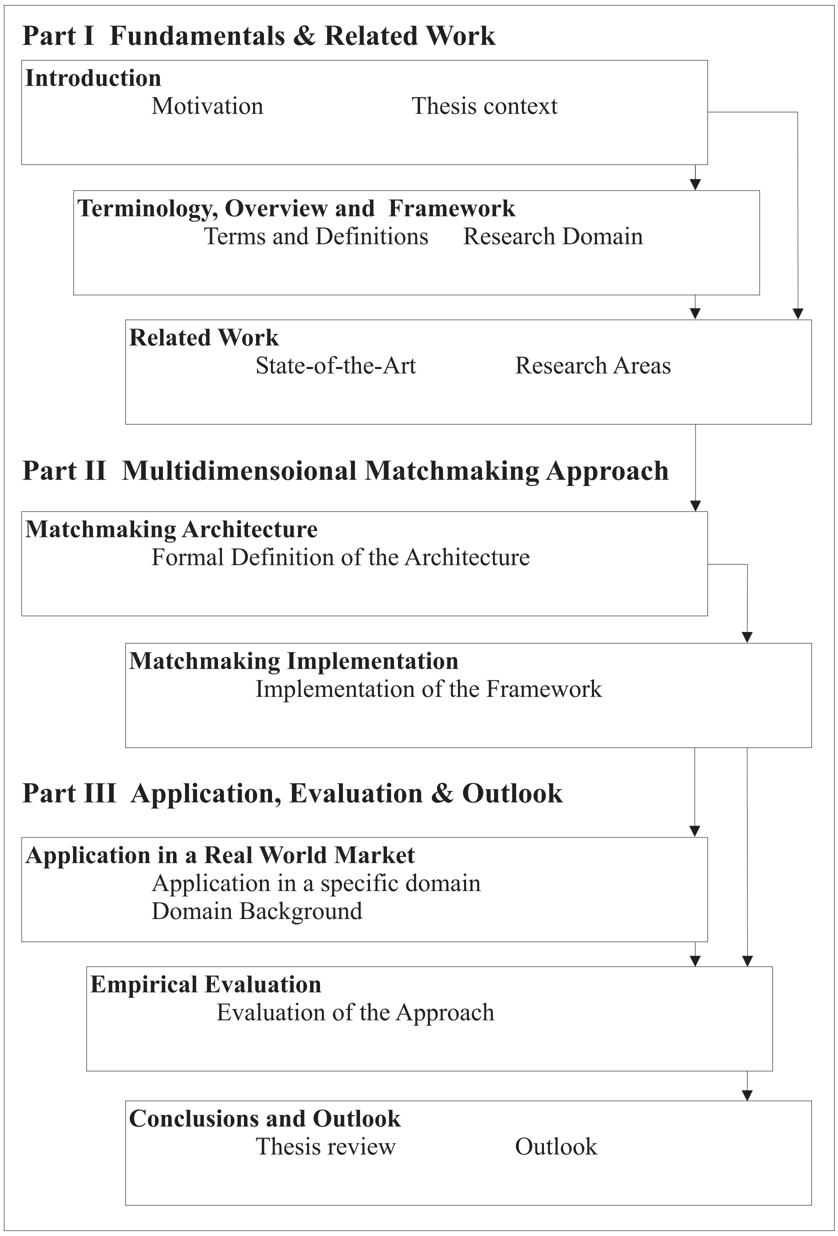

Part I Fundamentals & Related Work
Introduction
Motivation
Thesis context
Terminology, Overview and Framework
Terms and Definitions
Research Domain
Related Work
State-of-the-Art
Research Areas
Part II Multidimensoional Matchmaking Approach
Matchmaking Architecture
Formal Definition of the Architecture
Matchmaking Implementation
Implementation of the Framework
Part III Application, Evaluation & Outlook
Application in a Real World Market
Application in a specific domain
Domain Background
Empirical Evaluation
Evaluation of the Approach
Conclusions and Outlook
Thesis review
Outlook

Figure 1.1. Structure of this Work

1.2.2 Historical Perspective

The most important research results from the GRAPPA project have been published within the past two years in several articles. In Eiter et al.

(2001) the general understanding of multidimensional matchmaking as well as matchmaking for arbitrary objects is introduced. The focus is concentrated on matchmaking in electronic marketplaces in Veit et al. (2001). The application of multidimensional matchmaking mechanisms in electronic markets is analyzed in Veit et al. (2002b). Finally the GRAPPA framework is empirically evaluated in Veit et al. (2002a).

The work on hand incorporates the research results derived from the GRAPPA project so far and embeds them into a formal overall basis for matchmaking in electronic negotiations. Several important details have been published in the articles described above.

2 Terminology and Overview

The terms and definitions from the research areas of economics and computer science are not homogenous. The aim of this part of the work is to define a terminology which provides a basis for the definition of a multidimensional matchmaking framework.

In the economic research area negotiations and electronic negotiations within today's electronic markets are investigated. In computer science the areas of Information Retrieval (IR) and Distributed Artificial Intelligent (DAI) with the notion of Multi-Agent Systems (MAS) are concerned. It is the mission of this chapter to provide definitions from these areas which refer to each other and combine the potential of all three fields of research to provide a solution to an important problem within today's electronic markets.

The chapter is structured as follows: Focussing on the applied terminology in Section 2.1 as well as shaping the approach by defining the general field in Section 2.2 opens the chapter.

These topics are followed by Section 2.3 outlining the role of matchmaking in multidimensional electronic negotiations. Subsequent to this section the phases of the multidimensional matchmaking process are outlined.

These constituent contributions are followed by sections which deal with an overview on the applications of matchmaking: In Section 2.5 some domains for multi-attribute negotiations are introduced. These domains are outlined here and one of those, the domain of matchmaking in human resources negotiation, is discussed in more detail preparing Chapter 5.

2.1 Terminology

The central definitions and terms which are used in this work are presented here. The creation of the multidimensional matchmaking framework which has been developed within the last three years is driven by two main directions.

In Economic research a phase oriented sight is considered. Within this research area the concept of an electronic market and its phases play a central role for this work. There exist different approaches to the notion of an *electronic market*. This term is discussed in detail in the following.

Electronic negotiations play a very important role within today's applications of electronic markets. Research on electronic negotiations is a very important and up to date topic and currently engages a wide variety of researchers from many different disciplines. The definitions which lead to the term electronic negotiation as they are presented here focus, generally spoken, on the role that electronic negotiations play in markets. Electronic negotiations are understood as a mechanism to coordinate among offers and requests in economic markets.

2.1.1 Economic Concepts Used in This Work

The central economic concepts for this work are defined in a bottom up way. Starting from electronic media the concept of an electronic market and electronic negotiation is introduced.

As Ströbel and Weinhardt (2002) state in the Montreal Taxonomy for eNegotiations the notion of electronic negotiations is based on the definition of the term media and the media reference model (MRM, see Schmid 1998). In this context the term medium (pl. media) is defined as follows:

Definition 2.1.1 ***Media***

> Media are (software) platforms where interaction among entities is applied to coordinate the exchange of services or goods. An electronic medium (**eMedium**) is a medium of electronic (digital) channels which transport data.

Based on this definition of media the notion of a transaction can be introduced:

Definition 2.1.2 ***Transaction, Agent***

> A transaction is defined as the process of transferring one or more services or goods from one entity to another, called agents.

Here, entities are system participants. It is currently insignificant which kind of they are. Transactions are mostly not performed in a stand alone way. The term deal is introduced to enable clusters of transactions.

Definition 2.1.3 ***Deal***

> The clustering of a set of transactions which belong together, performed at a certain point in time by two or more entities on a medium constitute a deal.

In a deal at least two parties play a role. One is supporting a service or good and one is requesting it. These roles are specified in the following definition.

Definition 2.1.4 ***Offer, Request***

> An offer is a specification of an entity which is interested in supporting its capabilities, a good or service to another entity (counterpart) for a mutually calculated gain (win). A request defines a certain demand, specifying the capabilities, the good or service needed to fulfil the needs of the requesting agent. On eMedia, offers and requests are formalized in a machine readable and understandable way.

After defining what a deal is and how offers and requests are arranged into these concepts, now the notion of an electronic market can be introduced:

Definition 2.1.5 ***Electronic Market (eMarket)***

> If the deals, executed on the medium are economic goods, the medium is defined to be an electronic market (eMarket).

The foundations are now layed to define the understanding of electronic negotiations in this work. Before a deal can take place in an electronic market, the counterparts have to agree on the deal's content. This means that an offer and a request have to be specified and a commitment has to be agreed on. The requester commits on accepting the offer under the given conditions and the supplier commits on the conditions of the requester.

Definition 2.1.6 ***Electronic Negotiation (eNegotiation)***

> An electronic negotiation is an offer and request based communication and decision making process within an eMedium with at least one or more explicit and enforced rule(s), where at least one communication or decision-making task is assisted.

This definition of electronic negotiations in the sense it is used in this work is referring to the definition given by Ströbel and Weinhardt (2002) in the Montreal Taxonomy. More precisely, it covers the definition of electronic negotiations in the narrow sense. A wider sense of defining eNegotiations would include offer and request exchanging via an unstructured communication medium such as a chat room.

Now the term electronic auction is considered which is seen as a special type of eNegotiations. The following definition is related to the definitions given by Weinhardt (2002) and Bichler et al. (2002).

Definition 2.1.7 ***Electronic Auction (eAuction)***

> An auction is a mechanism to sell an object (good, service, etc.) under application of a bidding process with concurring bidders. The bidding process underlies a strict protocol which enables the participants only limited kinds of interaction.

EAuctions play a major role in todays electronic commerce applications on the Internet. The main reason is the fact that auction protocols are suited for easy implementation in programming languages due to the strictness of the auction protocols. Beam and Segev (1998) give an overview on the spread of usage of protocols for negotiations on the Internet. Hence, many classical auction protocols have been implemented in electronic commerce applications. Auction mechanisms and, consequently, eAuction implementations provide a narrow negotiation paradigm. The term eNegotiation aims on a wider field of negotiations to be represented in electronic media.

The domain of eNegotiations is an active and ongoing research area. Many different groups are involved into the definition, the development and the usage of eMedia for negotiation purposes. There is the Information Management and Systems domain in which currently the above mentioned Montreal Taxonomy is developed which classifies and analyses the aspects and issues which are important, defining eNegotiations.

In Chapter 3 research directions and concrete applications are discussed in more detail. In the following the key definitions for multidimensional matchmaking are presented.

2.1.2 Concepts from Computer Science

Two research issues are focussed here: Firstly, the notion of multidimensional matchmaking is introduced. Secondly, the domain of Multi-Agent Systems (MAS) and within those the notion of intelligent autonomous agents is defined. In this work MAS and intelligent autonomous agents are used as a modelling paradigm to realize multidimensional matchmaking mechanisms in electronic negotiations.

Terms and Definitions for Matchmaking The notion of matchmaking is introduced in a bottom up way. Starting from the common concept of an information object matchmaking is defined via several auxiliary concepts.

Definition 2.1.8 ***Atomic Type, Complex Type***

An atomic type is a well defined type which is semantically not further separable: E.g. integer, real, free text, identification number, time, date, etc.
A complex type is always compound from atomic types. The complex type constructors which are mainly used are

(i) *List:* A list of elements is a (possibly empty) row of elements which ends with a terminating symbol.
(ii) *Set, multiset:* A set of elements is an unordered collection of elements of the same type. A multiset is a set in which two or more elements can be identical.
(iii) *Record:* A record of elements is a consecutive row of a fixed number of n elements of possibly heterogeneous type.
(iv) *Array:* An array of elements is a consecutive row of a fixed number of n elements of the same type.

A complex type is defined by inserting one or more atomic types into a complex type constructor. The types can be cascaded by recursively calling one or more complex type constructors after one another.

Definition 2.1.9 ***Attribute, Dimension***

(1) An **attribute** is a concrete element of a structure, having a specific atomic type. Depending on semantics which the current application demands atomic types can be defined either strictly or in a more relaxed way. If the attribute is defined strictly, only instances of the specific atomic type are allowed as instances. If the relaxed definition is chosen, the semantics of this specific aspect of the description of the good or service are not binding, i.e. the attribute represents a container.
(2) A **dimension** is a field which can be addressed separately and consists of several attributes. Designing structures dimensions are created from attributes which are topologically belonging together. Also lists, records, sets, multisets or arrays, which are the main complex type constructors can be used to compose a dimension from attributes.

Definition 2.1.10 ***Multidimensional, Multi-attribute***

(1) With the term **multidimensional** the structures with more than one dimension are described. Dimensions can contain attributes or subdimensions themselves. Subdimensions can again consist of subdimensions or attributes.
(2) The term **multi-attribute** is weaker than the term multidimensional. Multi-attribute eNegotiations or multi-attribute match-

making is performed on more than one attribute. Hence a multi-attribute matchmaking or eNegotiation process is a process which performs a negotiation or distance computation on records or sets of attributes. In multi-attribute structures no substructures are allowed.

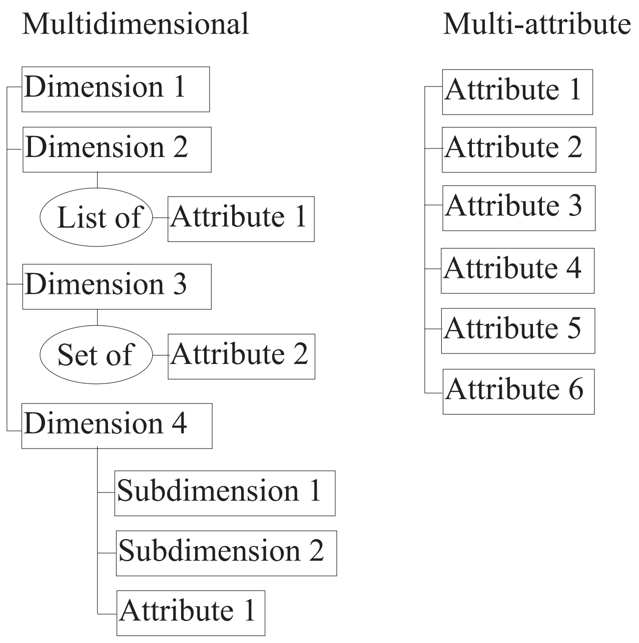


Figure 2.1. Example: Multidimensional and Multi-attribute Structure

In Figure 2.1 a simple example for both, multidimensional and multi-attribute negotiation object structures is given.

In applications an important issue is the semantic dependence or independence of attributes and dimensions. This issue – which corresponds to the definition of orthogonality in Mathematic contexts – is focussed in Chapter 4.

Definition 2.1.11 ***Information Object (IO)***

An information object is an object which specifies the information of an offer or a request. An information object can be composed from one or more attributes or dimensions. Consequently an information object is called multidimensional or multi-attribute. It has an *Information Object Structure* (IOS) which is the form describing the structure of the content which is encapsulated by a concrete object. This structure is well defined and can be instantiated by information objects.

Between an offer and a request information object a distance can be computed. This is done in a recursive procedure which computes a distance on attribute level and continues computing complex distances on dimension level. Finally, an overall distance is computed on information object level. This distance defines the distance or score the concrete offer information object has reached regarding the current request information object.

Definition 2.1.12 ***Attribute Distance Function, Complex Distance Function, overall Distance Function***

Generally a distance function is a function which takes as input two elements from their value domains and maps them to a distance value $v \in [0;\ 1)$. There are three types of distance functions considered in this work:

(i) **Attribute distance function:** An attribute distance function takes as input two attribute instances, one from the request and one from the offer, and computes a distance value $v_{att} \in [0;\ 1)$ which reflects the distance. This value reflects the distance of the attribute from the offer instance to the regarding attribute of the request instance.

(ii) **Complex distance function:** A complex distance function takes as input the results of all attribute distance functions which occur in a dimension. These are aggregated and a complex distance value $v_{com} \in [0;\ 1)$ is computed which reflects the distance of the dimension of the offer instance towards the current request instance.

(iii) **overall distance function:** Finally an overall distance function represents the function, which aggregates all dimension (or top level attribute) distance values to one overall distance value $v_{overall} \in [0;\ 1)$. This overall distance value represents the distance of the current offer information object towards the respective dimension of the request information object.

(ii) is not needed in case the considered scenario is a multi-attribute scenario. Then in (iii) the so called top level dimensions composed to the overall distance.

Now the foundations are layed to focus on the definition of matchmaking.

Definition 2.1.13 ***Matchmaking***

Let O be a set of offers and $r \in R$ be one request from a possible set of different requests. Matchmaking is seen as the task to compute the best fitting potential counterparts for a transaction execution. The matchmaking procedure computes

(i) an order

$$ord_r : O \times O \rightarrow \{-1, 0, 1\}$$

on O where for $o_1, o_2 \in O$: $ord(o_1, o_2) = -1$ means that o_1 is less relevant to r than o_2, $ord(o_1, o_2) = 0$ means that o_1 and o_2 are equally relevant to r and $ord(o_1, o_2) = 1$ means that o_1 is more relevant to r than o_2 as well as

(ii) a distance

$$rel_r : R \times O \rightarrow [0;\ 1)$$

for each tuple (r, o), i.e. one request $r \in R$ and one offer $o \in O$. The distance is a real value between 0 and 1 (distance value) and is computed under application of an overall distance function, recurring to atomic distance functions.

Overall, the matchmaking procedure takes as input a tuple (r, O), where r is one concrete request and O is a set of offers. The same tuple is given as output, where the set O is ordered, i.e., for each $o \in O$ a distance value rel_r^o is computed and assigned to the offer o. So the output tuple is defined as (r, O_{ranked}). Depending on the applications properties the roles of offers and requests can also be swapped (one offer matched against a set of requests).

This definition of matchmaking specifies the procedure which is performed semantically. It is a non trivial task to realize matchmaking in multi-attribute or even multidimensional scenarios. The challenge is to define and realize a matchmaking component which is configurable, implementing arbitrary multidimensional information object structures as well as supporting several distance functions for distance calculation. This is done in Grappa which is in detail introduced in Chapter 4 and Chapter 5. In Chapter 3 also the correlation between matchmaking and the matching phase in electronic negotiations is focussed.

The Domain of Multi-Agent Systems Software techniques and paradigms which are used conventionally for implementation of complex systems do not meet the demands which are needed to model generic matchmaking processes. In matchmaking processes two or more counterparts take part. To model an electronic market efficiently it is straightforward to assign each offer and each request to an autonomously acting market participant. Each offer and each request should be represented by a software entity which knows about the properties and is able to represent the will of its owner.

Definition 2.1.14 ***Properties, Will***

(1) The **properties** of the owner of an offer or a request are the offer and request specifications which are determined in the offer and request representations.
(2) The **will** of the owner is defined within the concrete offer or request instance by specifying negotiation thresholds for each dimension or attribute as well as importance weights. These values indicate the direction in which the offer can be degraded or upgraded in the view of the owner.

When an owner of a software entity which represents an offer or a request inserts his or her specification into an electronic market, this owner is called market participant.

The properties of the offers and requests can be mapped into an information object structure which is designed for a market application. Another important issue is the scaleability. An electronic market approach should be able to deal with few, more or even a huge amount of offers and requests.

Multi-Agent Systems (MAS) are based on a paradigm which is developed in the research domain of Distributed Artificial Intelligence (DAI) and meets the desiderata listed above. The research domain of MAS has been focussed by many different groups in various fields. The definition of a software agent – one autonomously acting unit in a MAS – has been discussed in depth in literature but no uniform definition has been agreed on. Wooldridge and Jennings (1995) and Wooldridge et al. (1996) give a definition which is used in wide areas of DAI. The Foundation of Intelligent Physical Agents (FIPA) also presents a definition in the FIPA 1997 specification FIPA Members (1997). Further specifications have been proposed. Also Subrahmanian et al. (2000) present a related view of heterogeneous agents which communicate using a uniform language.

Definition 2.1.15 ***Agent Communication Language (ACL)***

An agent communication language is a language with precisely defined syntax, semantics and pragmatics that is the basis of communication between independently designed and developed software agents.

The most prominent example for an ACL is the FIPA-ACL defined in depth in the current FIPA standard. Based on the properties and definitions given above the notion of a software agent with respect to its application in this work is defined. See Wooldridge and Jennings (1995) and Weiss (1999) (p.32).

Definition 2.1.16 ***Software Agent***

> A software agent is a piece of software which has the following properties:
>
> - **Autonomy:** A software agent should be able to fulfill the majority of their tasks without explicit permission of its owner. It should control, decide and execute its tasks without further communication with its owner, i.e. autonomously.
> - **Proactiveness/Consciousness:** A software agent should be proactive, i.e. it should be able to initiate an action in order to solve the tasks it is designed to.
> - **Reactiveness:** A software agent should be able to respond, based on its knowledge and cooperation model on actions of other agents participating in the system. An agent should be able to recognize changes in its environment and take these into its consideration for future actions.
> - **Social behavior:** A software agent should be able to communicate, interact and cooperate with other software agents in order to solve its tasks. These communication and interaction processes include cooperative and competitive behavior.

The first three of these properties describe the properties of a single agent. Consequently it is possible to implement those more or less directly. The fourth property is a property which focuses on the entire Multi-Agent System. Wooldridge and Jennings (1995) describe this property as a *weak notion of agency* which has to be made explicit.

Definition 2.1.17 ***Multi-Agent System (MAS)***

> A Multi-Agent System is a software system which consists of a set of software agents (two or more) exchanging goals (solution of a task) for a mutual beneficial cooperation. The (autonomous) agents which are the elements of a MAS can have different degrees of autonomy.

The degree of the autonomy of an agent depends on the amount of decision space the agent's owner is providing to its agent. An agent has only a low degree of autonomy when the user has to decide each step, before the agent is allowed to commit to another agent. Whereas an agent has a high degree of autonomy if the user passes on a big amount of decision capability to its agent (see Faratin 2000). Furthermore it is distinguished between open and closed MAS.

Definition 2.1.18 ***Closed MAS***

> A *closed* MAS is referred to as a system which is characterized by a central designer undertaking the following steps in the system design methodology:
> 1. Definition of the global problem(s).
> 2. Mapping and assigning subproblems and resources to software agents, either dynamically at run-time or statically at design-time.
> 3. Central configuration of all agents, specifying their agent's behaviors in ongoing interactions with other software agents.
> 4. Using an agent communication language (ACL) to allow the agents to solve the problems in step 1.

A closed agent system is also referred to as a distributed problem solving (DPS) system (see Durfee and Rosenschein 1994). This methodology is problem centered and refers to an approach, applying agent technology for the solution of problems which usually have been solved in centered systems before.

In this context the problem to be solved is specified in step 1. A central designer creates a fixed and static society of computational agents in step 2, who interact repeatedly (exchanging subproblems, plans and information) using the ACL in step 4. In this manner, they collectively solve a well structured and objective global problem. Here, agents are often homogeneous in architecture, languages and reasoning, see step 3, and act cooperatively, i.e. they are motivated to help one another to solve the global problem.

Definition 2.1.19 ***Open MAS***

> An *open* MAS is a Multi-Agent System, which is characterized by a number of designers undertaking the following steps in the systems design methodology (see also Faratin 2000):
> 1. Either define a global problem or allowing the problem to dynamically emerge.
> 2. Nominating/selecting (pre-existing) autonomous agents to enter interactions.
> 3. Configuration of the agents the designers want to use.
> 4. Using an agent communication language (ACL) to allow the agents to identify conflicting issues and solve problems in step 1.

Open MAS are also characterized as encounters, where pre-existing agents come together infrequently to solve a problem, trade goods, commitments or services, or, alternatively where problems emerge dynamically in the course of interactions. This interaction is centered to the individual agents' benefit which makes one main difference towards closed agent systems where the individual agents subordinate to the solution of the common problem. Any agent is free to come and go. It is also possible for external entities to cooperate with heterogeneous software agents, which are designed using different software platforms. Coordination in heterogeneous software agent applications is studied in detail by Subrahmanian et al. (2000).

Now specific kinds of agents in open MAS are introduced. In an eMarket there are different roles which have to be covered. On the one hand there are the participating parties represented by software agents as defined above. These agents communicate on a platform using a common ACL. In order to implement an efficient eMarket using software agents certain agents, who are not taking part in the actual eMarkets business are needed to mediate between the market participants. This leads to the notion of middle agents, see also Decker et al. (1997). The term middle agent is an umbrella term referencing different types of agents playing different roles.

Definition 2.1.20 ***Middle Agent***

A **middle agent** is an agent which mediates between offering and requesting agents. There are several different middle agent roles possible.

Definition 2.1.21 ***Broker Agent, Matchmaker Agent and Blackboard Agent***

See also Figure 2.2.

(1) **Broker agents** collect both, offers and requests. A broker agent plays two roles: It represents the centroid for all candidates as well as the candidate for all centroids.
(2) **Matchmaker agents** collect offers. A matchmaker agent only plays one fixed role. It collects offers and provides ranked lists of appropriate offers to centroids.
(3) **Blackboard agents** collect requests. A blackboard agent works inversely to the matchmaker agent and, therefore, also plays only one fixed role.

Definition 2.1.22 ***Centroid, Candidate***

(1) A **centroid** is defined as an agent carrying a request.
(2) A **candidate** is defined as an agent carrying an offer.

This definition is chosen due to the fact that in the matchmaking applications in this work, always $1 : n$ matchmaking processes are considered. Consequently, the centroid is always a single representant of one matchmaking side.

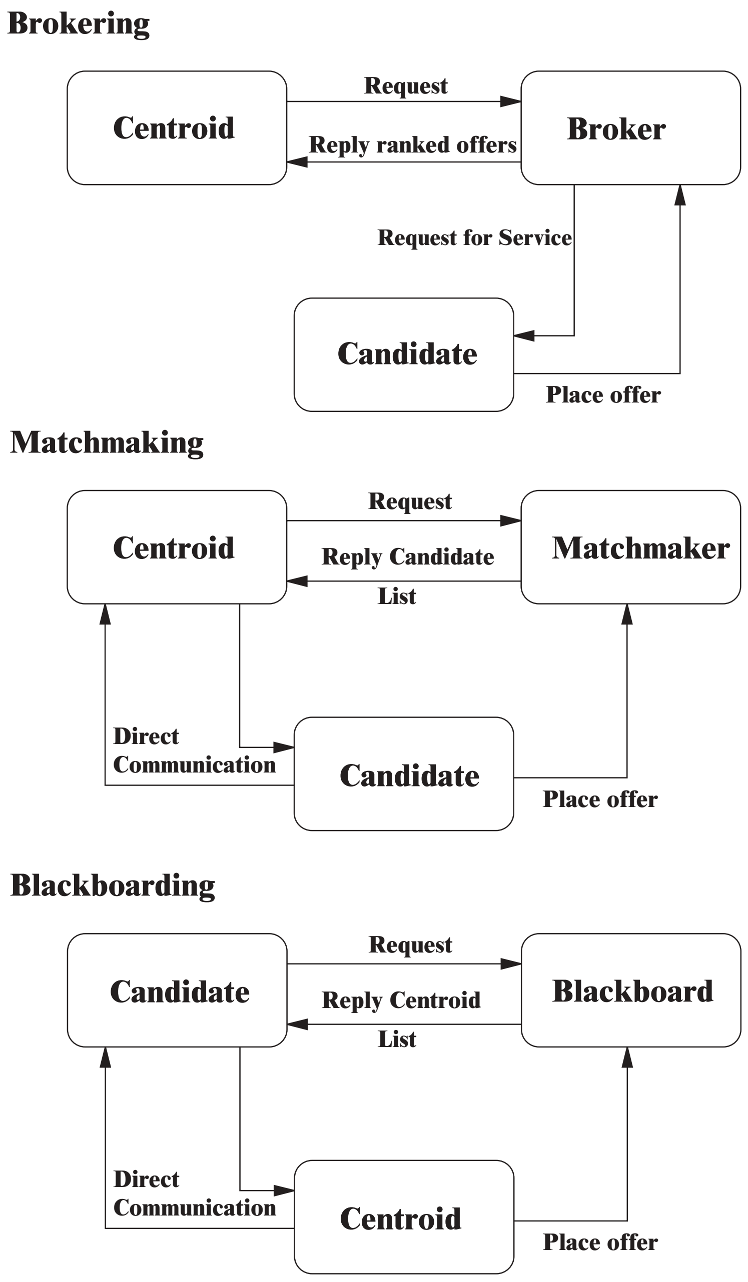


Figure 2.2. Brokering, Matchmaking and Blackboarding

Obviously, all three middle agent approaches from above hold certain advantages and disadvantages. Brokers provide a maximum in privacy: The centroid and the candidates are not known to each other because all communication is running over the broker. On the other hand the broker represents a communication bottleneck. Matchmakers on the other hand are well suited for markets providing high liquidity on the offer side. They are in this way computationally more efficient but provide only limited privacy.

Blackboards are on the other hand well suited for markets providing high liquidity on the request side. But also provide only little privacy.

In the economic sense, agents are defined as entities in market scenarios. Regarding the definitions from Section 2.1.1 in transactions at least two counterparts play a role. These counterparts are defined as agents.

Definition 2.1.23 ***Agent***

> An agent is defined as an acting entity in the market scenario. This might e.g. be a human negotiator, an organization or an autonomous software agent. It is the fundamental actor in a domain. It combines one or more service capabilities or encapsulations of goods or services into a unified and integrated execution model which can include access to external software, human users and communication facilities.

In this work, the role of an agent in a market scenario – defined in the economic sense – is mapped to a software agent to represent market participants in a partially automated electronic medium.

2.2 General Field

In many areas of electronic information processing matchmaking plays a crucial role. The term matchmaking often addresses different procedures.

In the last section detailed definitions were provided which are now used to integrate the heterogeneous disciplines from economics and computer science to this work.

In the context of electronic negotiations, the term matchmaking is used when a request and an offer of a service, a good or a property are compared using automated techniques. These techniques vary strongly from domain to domain. The matchmaking procedure is mostly divided into many steps, subprocedures and uses several tools. In this work many of these techniques applied in matchmaking are introduced. The main application domain is the modelling of eMarkets using the paradigm of Multi-Agent Systems (MAS). Within eMarkets a strong focus are multidimensional electronic negotiations. This special type of negotiation which has recently gained interest in the research community is illuminated in detail in this work. The matchmaking concepts developed in this research contribution are not exclusively bound to Multi-Agent Systems. The mechanisms can be applied to any computational domain.

As an application on which the capabilities of the matchmaking approach will be demonstrated, the prototype implementation of a generic multidimensional matchmaking component is introduced. This prototype implements the theoretic multidimensional matchmaking framework GRAPPA in a JAVA package.

2.2.1 Electronic Markets

As to Schmid (1993) electronic markets are defined following three phases. When combined, these phases define an electronic market.

Definition 2.2.24 ***Electronic Market***

An electronic market is a coordination mechanism by which a transaction is supported in three phases:

(i) **Information and Approach phase:** In this phase possible transactions are approached. Possible actors exchange information on the goods, services or contracts which are offered or requested.

(ii) **Intention and Agreement phase:** In this phase offers and requests of the market participants are coordinated and the conditions are negotiated.

(iii) **Clearing and Settlement phase:** Finally, the transfer of services or goods is performed according to the negotiated conditions.

This formalization of electronic markets is very open and allows a wide range of technologies and methodologies to be applied for coordination among offers and requests. Firstly it is differentiated which technologies can be applied in electronic markets, regarding this definition.

An electronic market is a system based on information technology which covers all or at least one phase of the market transaction process. From these definitions some relevant properties of electronic markets can be derived.

- Because of the application of information technology in electronic markets neither goods (traded items) nor actors have to be geographically concentrated.
- Electronic markets assist the participants in exchanging digital goods, services or, more general, property rights and to lead in an efficient way to a conclusion of a contract.

In traditional markets there are numerous sub-problems which are solved by humans and applied to make the market work properly. Switching to the paradigm of electronic markets makes it necessary to solve these sub-problems in electronic procedures.

Many of these problems can be addressed straightforward. Communications which take place face to face among humans can be mapped to electronic networks. Information about offers and requests can be stored on highly efficient storage media.

But there remain sub-problems which are hard to formalize and solve in electronic markets. One of these is the coordination among offers and requests. The two major coordination problems which have to be considered are

(i) The selection of (a subset) of all available offers for a request with whom a negotiation would be possible (in the information and approach phase).
(ii) The automatic comparison of offers and requests to determine the best offer for a given request in each step of a negotiation (in the intention and agreement phase).

Both of these problems are addressed with the matchmaking approach in this work.

2.2.2 Electronic Negotiations and Electronic Auctions

The intention and agreement phase in electronic markets is addressed by different market coordination mechanisms. The main roles play electronic negotiations and electronic auctions. In current electronic market approaches electronic auctions are mainly applied for the determination of the counterparts of a contract.

But auctions and so electronic auctions have certain disadvantages and limitations. One main problem is the reduction to price. While it is easy to implement and formalize auction mechanisms for single-attribute applications like in most scenarios where highly standardized products are traded, where only the price has to be negotiated, it is hard and computationally costly to formalize auction mechanisms for multi-attribute or even multidimensional domains. E.g. applications in which attributes besides the price have to be agreed on.

While electronic auctions are applied in practice since several years, electronic negotiations research is at the very beginning of intense research. They try to implement real life negotiations applying programming paradigms. Regarding the view of this work electronic auctions ar a subfield of electronic negotiations, i.e., every electronic auction is an electronic negotiation but not vice versa. This is also shown in Figure 2.3.

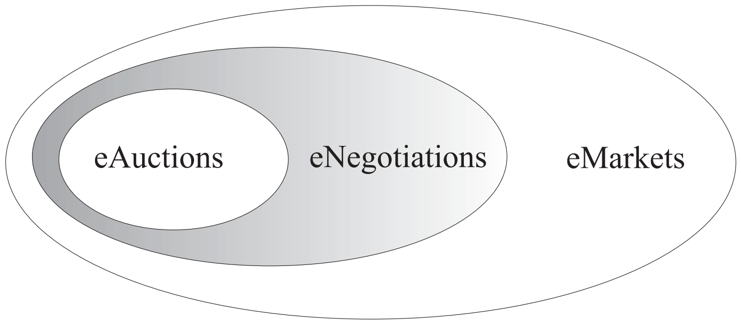


Figure 2.3. Electronic Auctions, Electronic Negotiations and Electronic Markets

2.2.3 Multi-Agent Systems

Within coordination approaches for Multi-Agent Systems in computer science several concrete matchmaking approaches have been proposed so far. These approaches are studied in depth in this work in order to derive some of the expertise which has been gained in this field. However, the approach presented in this work exceeds the known approaches in several areas. Most approaches presented in literature, like Sycara et al. (1999) or Subrahmanian et al. (2000) and Arisha et al. (1999) focus on matchmaking procedures which have a given, non-flexible object structure and are designed for a specific problem.

The most important aspect is to differentiate in which application level the matchmaking procedure is applied in an agent system. Agent based approaches can be divided into approaches where agents are used as implementation technique as well as into approaches where the properties of software agents are used to incorporate business logic. Corresponding to these two directions, matchmaking within agent systems can be seen in two different ways.

(1) In MAS, on the *one hand* matchmaking can be used for the comparison of service descriptions of agents. This type of matchmaking is a task which is integrated into the technical foundation of Multi-Agent Systems. Agents can find other agents via their service description which every agent carries. This matchmaking enables distributed problem solving using services of other agents. An approach which is tending towards this direction is presented by Sycara et al. (1998) in their approach *Language for Advertisement and Request for Knowledge Sharing* LARKS.
(2) On the *other hand* matchmaking in agent systems can be used to perform a determination of distance among profiles which contain application and user specific information. This matchmaking enables the application of the Multi-Agent Systems paradigm for solving real-world problems such as finding the right offer for a request in a market scenario. Using

this type of matchmaking the programming paradigm of Multi-Agent Systems is only used as a tool to model market coordination mechanisms with heterogeneous market participants. The focus here is not on the internal coordination among the individuals in the Multi-Agent System itself, but on the application of the multi-agent paradigm for real-world problem solving, e.g. market coordination. An approach which shows the potential of multi-agent technology for this issue is given by Weinhardt and Gomber (1999) with their agent mediated trading approach *Agent Mediated TRAding System* AMTRAS.

Of course, a real-world scenario must not use any software platform as a pure service for problem solution yet. System dependent properties are immanent to every programming paradigm. For this reason it is not possible to choose one of the extremes mentioned above. Actual implementation paradigms always tend between these two alternatives and try to be as close as possible both aspects (1) and (2). The main intention which is focussed on is to avoid system immanent factors which influence the proper and system independent logic of the application which is intended to build.

In the context of multidimensional matchmaking the multi-agent paradigm is used to encapsulate the offers and requests of market participants. Each entity of a market is wrapped by an agent carrying the information the owner is willing to release to participate in the market. The matchmaking procedure involves functions which are designed to mirror human judgement of an expert as realistic as possible.

2.3 Matchmaking in Electronic Negotiations

eNegotiations are executed in the intention and agreement phase of an eMarket. In Figure 2.4 the authors Ströbel and Weinhardt (2002) (p. 8) show the phases an eNegotiation process can be divided into. To define the term matchmaking in eNegotiations more precisely the definitions of Ströbel and Weinhardt (2002) are adopted to the generic matchmaking model. The single steps inside the intention phase have the following characteristics:

- **Offer and request specification:** The agents have to specify offers and requests indicating their constraints and preferences towards the transaction object. This specification may also include the provision of signatures or the definition of timestamps (to express validity). This specification can be executed instancing a centroid or a candidate information object which has been designed for this specific market.
- **Offer and request submission:** To submit an offer or a request, the agent can actively send the offer or request to a specific agent (middle agent) or just notify the middle agent of the completion of the specification.

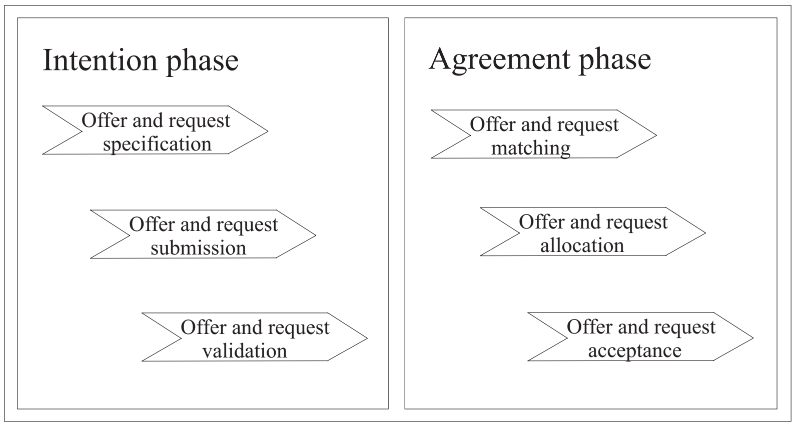

Intention phase
Offer and request specification
Offer and request submission
Offer and request validation
Agreement phase
Offer and request matching
Offer and request allocation
Offer and request acceptance

Figure 2.4. Intention and Agreement Phase

- **Offer validation:** When the middle agent receives the offer or request, the information object is checked for completeness and the compliance with certain rules.

The single steps in the agreement phase have the following characteristics:

- **Offer and request matching:** The aim of this phase is to find pairs of offers and requests which stratify potential conterparts for a transaction execution. This includes the identification of all offers which match a given request. The matching phase may also include a scoring procedure. The aim of the scoring procedure is to identify the best matching offer for a request. In the matchmaking framework in this phase a ranking of all offers with respect of the current request is computed and returned as a ranked list to the centroid (requesting agent).
- **Offer and request allocation:** In this task the counterparts for a possible transaction are determined using the information from the matching and scoring phase. The duties of the single agents are determined and assigned. If the selected offers and requests still feature certain value ranges or options the final configuration has to be determined in this phase.
- **Offer and request acceptance:** This final task serves the acceptance of the terms and conditions which have been determined above. The agents have to accept the conditions in order to execute the transactions and complete the deal.

The generic multidimensional matchmaking mechanism which is developed in this work to cover the matching and scoring phase in eNegotiations. The Grappa framework is defined and the implementation described in Chapter 4 and Chapter 5. In these chapters a generic and adaptive matchmaking component which is easily integrated into any eNegotiation framework is provided.

2.4 Phases of Matchmaking

The matchmaking process itself clusters into two phases: the definition and the execution phase. Each phase is divided into three steps again. The idea in the definition phase in multidimensional matchmaking is that the designer of the application creates the matchmaking procedure using a generic toolset. Firstly, offer and request structures have to be specified. This is done by determining the attributes which describe the offer and the request best. These attributes have to be clustered into dimensions which belong together semantically. The dimensions and – in case there are – subdimensions have to be composed to an overall structure of the request and the offer information object. Then, a mapping between the offer and the request has to be defined. This is due to the fact that offer and request may be of heterogeneous structure. Hence, it might be not the identity mapping offer to request structure. In this procedure also the distance functions have to be specified, which will be applied to determine the distance among the instances of the atomic and complex types.
The single steps within the definition phase of multidimensional matchmaking (see Figure 2.5) are defined as follows:

- **Identification of matchmaking attributes:** In this step the attributes, which describe the objects to be offered and requested, are identified. Atomic types are assigned to the attributes. Value domains are created.
- **Clustering of attributes to dimensions:** The attributes which have been identified in the last step are clustered to dimensions. This is done separately for the request and the offer structure. The clusters should contain attributes which belong semantically together. Here, the orthogonality and preferential independence among the attribute clusters are important. This issue is treated in Section 4.3.
- **Mapping between offer and request dimensions:** The offer and request structure may be heterogeneous. For this reason a mapping has to be defined which covers the semantic structure of the offer and the request. The mapping has the following properties:
  - (i) It maps each set of dimensions of the offer to a set of dimensions of the request. Note: the sets can possibly contain only one dimension.
  - (ii) It is surjective and complete, i.e., it covers all dimensions of the request and the offer. Consequently, no information which has been provided in the offer or the request is omitted.
  - (iii) It contains each dimension and each attribute of the offer and the request structure. For each attribute pair – one attribute from the offer structure and one attribute from the request structure – an *atomic distance function* is specified in the mapping. Also for each pair of dimension sets (see (i)) a *complex distance function* is specified.

Now the single steps in the execution phase of multidimensional matchmaking also shown in Figure 2.5, are described:

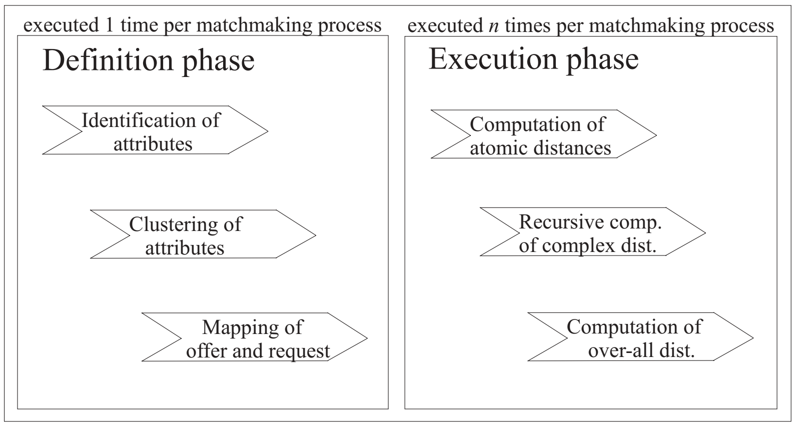

executed 1 time per matchmaking process
Definition phase
Identification of attributes
Clustering of attributes
Mapping of offer and request
executed *n* times per matchmaking process
Execution phase
Computation of atomic distances
Recursive comp. of complex dist.
Computation of over-all dist.

Figure 2.5. Phases in Multidimensional Matchmaking – Definition and Execution

- **Computation of atomic distances:** In this step the atomic elements of the request and offer information object are considered. The distances among the instances of the attributes are computed using the atomic distance functions which are assigned in the mapping defined between the offer and request structure.
- **Recursive aggregation of dimension distances:** The atomic distances from the last step are the basis for the aggregation to the dimension distances. For this step complex distance functions are applied which compute aggregated distances between the dimensions. Dimension distance values represent a lower degree of information compared to the sum of atomic distance values. The amount of information lost in the aggregation process depends on the choice of the so called *complex distance function* applied. Concrete examples of complex distance functions such as weighted average or hausdorff distance are given in Section 4.5.
- **Computation of overall distance:** Finally an overall distance value is computed using a complex distance function. This distance value represents the distance of one complete complex offer to a given complex request. The same complex distance functions like for dimension distances can be applied here. Different thoughts such as the completeness of information and k.o.-criteria play an important role for the choice of the overall distance function applied. In Section 4.5, also different approaches to this subproblem are proposed.

These phases depict the matchmaking application in arbitrary scenarios. In eNegotiations the designer of a negotiation framework can arbitrarily define the structures of the offers and requests. Of course, the offer and request structure must represent the application logic.

In the next section, application domains for eNegotiations are investigated. Special attention is payed to the domain in which the multidimensional matchmaking component is related to the human resources domain.

2.5 Application Domains for eNegotiations

As shown in the last sections, multidimensional matchmaking is a central task in electronic negotiations. As the Internet and electronic coordination mechanisms are continuously gaining importance in the modern economy, eNegotiations begin to establish in many domains.

In procurement and financial markets eNegotiations have already succeeded in major projects. Many companies switched their entire procurement processes to eMedia and many financial marketplaces are run on these media.

Scientists and practioners are actually making significant effort to formalize and implement eNegotiation scenarios enhance the range of transactions being possible enormously. The eMediator prototype by Sandholm (2000) is, e.g., an up-to-date eNegotiation prototype which implements negotiation concepts far beyond the common auction protocols, such as combinatorial auction protocols. Relevant contributions within this domain are also provided by Lo and Kersten (1999) with the INSPIRE system as well as the contributions of Schoop (2002).

In many different domains such as the trading of derivative financial instruments (see Bichler 2000a), the energy market domain (see Strecker and Weinhardt 2001) or the human resources domain (see Maier et al. 2000), first approaches have been made to electronize the markets and integrate eNegotiation mechanisms. Within the finacial market domain the AMTRAS approach (see Weinhardt and Gomber 1999) and the *Virtual Trading Room* (VTR) approach (see Budimir and Holtmann 2001) are to be mentioned.

There is a central problem which is common to all current eNegotiation approaches. While it is wide-spread to define strict trading rules for homogenous goods it is much harder to formalize abstract negotiation protocols to apply on heterogeneous goods as well as on personal needs and wishes. Actual eNegotiation approaches often use a highly domain dependent terminology and negotiation mechanisms.

2.5.1 Matchmaking in Multi-attribute Auctions

Matchmaking plays a central role within multi-attribute electronic auctions. In the definition of an electronic auction which is given in Section 2.1.1 the notion of an auction in the economic sense is referred to. Bichler et al. (1999) focus on this issue as follows: "An auction is an economic mechanism for determining the price of an item. It requires a methodology which is announced beforehand, one or several bidders who are interested in the item and the aim to sell the item to the highest bidder." (see Bichler et al. 1999, p. 155).

As also shown in the phases of an intention and agreement phase of an electronic negotiation, an auction mechanism also checks a bid for its validity and updates the active bids by the highest one. There are several different auction protocols. These are further considered in Section 3 in more detail. In this section, the meaning of matchmaking within electronic auctions is considered.

Multi-attribute Utility Computation A central role in electronic auctions and, generally spoken, electronic negotiations play functions which determine the value of an item for the bidder, i.e. the party requesting an item. The function is called utility function and takes as input the properties of the supplied items. These properties are mapped to a score which indicates the value of the item to the bidder, i.e. requester. In case of single-attribute auctions this is obvious. The utility function takes as input the item's attribute instance and maps this to a score.

In case of a multi-attribute scenario this is not as trivial. In this case a utility function must map each property of an offered item to a virtual currency or score and combine the virtual currency or score results of each attribute to an overall value. This overall value is called the overall utility of an item for a requester. Finally this overall utility decides which item wins. The item i.e. the candidate with the highest overall utility for the centroid should win due to the Pareto efficiency. An allocation is called Pareto efficient if it is impossible, by using them differently, to makye any one centroid or candidate better off without making at least one other centroid or candidate worse off (see also Lipsey 1989)

Definition 2.5.25 ***overall Utility Function Unified Weight***

An easy and straightforward way to express the overall utility function is the following:

$$U(x_j) = \sum_{i=1}^{n} U_i(x_j^i)$$

Where $U_i(x)$ indicates the utility function for the individual attribute i of an item j and $x_j = (x_j^1, x_j^2, \ldots, x_j^n)$, $j \in \{1, \ldots, m\}$ indicates the instance vector of attribute instances which item j supplies.

The result of this overall utility function is a non weighted sum of the attribute utilities. For practical application it is useful to combine this overall utility function with weights for the attributes. In this case, each attribute utility function is weighted with a real number $w_i \in [0;\ 1)$ before summing up to an overall utility. This sum of weighted attribute utility functions is divided by the sum of all weights for normalization.

Definition 2.5.26 ***overall Utility Function Distributed Weight***

The following overall utility function is obtained:

$$U(x_j) = \frac{\sum_{i=1}^{n} w_i U_i(x_j^i)}{\sum_{i=1}^{n} w_i}$$

Ideally the weights w_i for $i = 1, \ldots, n$ can be modified by each requester. So each requester is able to indicate its own preferences to an attribute.

The aim is now to allocate the deal to the suppliers of offers by maximizing the overall utility of the requester. Finally the supplier with the highest overall utility for the requester wins the auction:

$$\max U(x_j)$$

Where $j \in \{1, 2, \ldots, m\}$. If several offers reach the identical overall utility for a request they are treated indifferent and can be substituted by one another. These functions take as input an attribute instance and provide a value in the score or virtual currency value domain. This value must represent the preference of the requester best possibly. Consequently it is useful to let the requesters define their attribute utility functions individually.

Now, requesters have two possibilities of influencing the result of the overall utility computation. The first is the definition of the attributes' weights. This measure is made to assign the importance to an attribute in comparison to the other attributes. The second possibility consists of the determination of the attribute utility function itself. This would be possible if the attribute has a discrete value domain. An example for such an attribute would be the color of an item. A requester can easily define a personalized attribute utility function for such an item. Here, the idea is to assign a personal score to each possible color.

Matchmaking for Utility Computation Some authors started to work on the domain of multi-attribute utility computation for multi-attribute electronic auctions.

In the application scenario of multi-attribute electronic auctions the term matchmaking is defined and understood in this work as follows: According to the definition from Section 2.1.2 matchmaking is a procedure of identifying the offers which are closest to a request. It is straightforward to apply the matchmaking approach to multi-attribute electronic auctions. The attribute utility functions U_i are represented by the attribute distance functions defined for matchmaking. The overall utility function, which is the sum or the weighted sum of the single attribute utility function values is represented by the overall distance function within the matchmaking approach. The score which is used to determine quality measures as results from attribute utility functions is represented by the real values in the interval $[0; 1)$.

In the next section the application of multidimensional matchmaking in electronic human resources markets is described. In these eMarkets the implementation and evaluation of the generic multidimensional matchmaking component GRAPPA is done.

2.5.2 Multidimensional Matchmaking in Human Resources Negotiation

The human resources market is a central field in economics and management science. Researchers from Operations Research (OR), microEconomics, sociology and many other fields have been investigating this topic in depth (see Maier and Gollitscher 2001 for a collection of views from different research areas). Different skill matching methods have been developed. Also on the Internet a number of job portals have been established. E.g. in Monster[1] or Stepstone[2] it is possible for the applicants to submit a resume and to establish a request which searches the job databases in regular time intervals. However, the procedures available for matching the own resume against vacant positions are fairly rude and can hardly be personalized. As far as human resources marketplaces have been evaluated for this work the procedures are mainly based on matching keywords which are supported by the applicant against the given positions and vice versa.

Though the current applications for the electronization of the human resources market are simple, it takes intense effort to increase the quality of such applications. Like in many other electronic market domains, valuable information is lost during this process. When a person applies for a job not only the measurable factors such as skills, work experience, languages, degrees, etc. are important. In contrast, often persons are judged as qualified for a position by their personality or their character. Additionally, at an application also the form and structure of the domain play a central role. Factors such as correctness of the style, to make the right impression and last but not least the outfit of the person play an important role.

All these factors are *soft factors* which cannot or only partially be measured using mathematic or electronizable distance functions. Also the approach which is provided in this work does not support these features but still, much better matchmaking results compared to the actual methods are achieved

[1] http://www.monster.com (06/25/2002)
[2] http://www.stepstone.com (06/25/2002)

(see also Chapter 7). Consequently the computation of distances between persons who are looking for a job can only be modelled by taking the factors that are at least partially measurable. These factors are

- skills,
- work experience,
- free text description,
- salary,
- entry date,
- language skills.

These factors are modelled as dimensions of a request and an offer information object structure.

A profile structure for vacant positions as well as for the applicants is derived. The information object structures are now equipped with the atomic and complex distance functions. To perform matchmaking centroids and candidates the generic multidimensional matchmaking framework which is developed in this work is applied. The matchmaking procedure which is executed on the framework covers the matching section within the intention and agreement phase in a human resources market.

This approach does not yet go to an extent to tackle soft factors in human resources matchmaking; however, it is shown that the results achieved are considerably better than those of existing methods.

2.6 Summary

In this chapter the terminological basis for this work is defined. The key concepts from the disciplines in computer science and economics which play a major role for the research are examined. In Section 2.1 the terminological aspects are focussed. In Section 2.2 the terms and notions are integrated into the relevant reseach fields in economics and computer science. The key notions of "electronic negotiation" as well as "software agent" and "multidimensional matchmaking" are framed by application directions in current research.

Section 2.3 combines the electronic negotiation research with multidimensional matchmaking and gives evidence on the application of matchmaking mechanisms in this very actual research domain. In Section 2.4 the matchmaking procedure is presented as a phase oriented process which provides mechanisms to apply matchmaking in electronic markets.

Finally in Section 2.5 different real-world application domains for multidimensional matchmaking are discussed and introduced. The human resources domain is selected as central domain for this work.

The contribution of this chapter is to

- presenting an overall-view for the work, including a brief presentation of the Grappa framework.
- defining the key terminology and research fields, the work is influenced by.

In the next Chapter 3 the related work which is relevant for the work is introduced. Here, special focus in spent to concrete research projects and – especially in the matchmaking field – to concrete matchmaking algorithms.

3 Related Work

Since electronic business has become an integral component of today's economic processes, many electronic markets have been established and distinguished (see Neumann et al. 2002). The need for further automation of procurement, logistics, human resources planning, finance and many other domains is obvious. Scientific approaches for the automation of these processes are more and more applied in practice. In this chapter two main pillars which lead towards a stronger automation of electronic business processes are introduced:

- Matchmaking
- Electronic Negotiation

The aim of this chapter is to provide an overview on several research activities in electronic negotiation and matchmaking.

The chapter is structured as follows: In Section 3.1 the most prominent matchmaking approaches are outlined. Some contributions are presented in high detail because of their relevance to the Grappa project. This enables an easier comparison with the new approach presented in this work. In Section 3.2 several negotiation approaches are outlined. There exists a high number of negotiation approaches in different scientific communities. Hence, most approaches are introduced in less detail compared to the matchmaking approaches. In Section 3.3 the matchmaking approaches and the negotiation approaches are briefly compared. The results motivate the definition of the Grappa framework.

The content of this chapter is derived from the cited publications. The validity of definitions, examples and theorems is often limited to the concrete approach and not valid for the view in this work. The theoretic model and implementation framework which is developed in this work is based on the definitions given in Chapter 2. The views of other researchers who are working in the area of electronic negotiation and matchmaking and which are considered to be relevant for this work are presented here. The focus is to broaden the reader's understanding of the questions arising in electronic negotiation and matchmaking.

3.1 Matchmaking Approaches

In this section several existing "pure matchmaking approaches" are introduced. These systems focus on providing matchmaking mechanisms and provide no or only little negotiation support.

The notion of "matchmaking" is influenced by the idea of the "mediator" which has been introduced by Wiederhold (1993) and consecutive work. Several approaches which are proposed here – especially the approach of Subrahmanian et al. (2000) references Wiederhold's work.

In Section 3.1.4 preparing Section 3.1.5 the Sycara and coauthors' sight on matchmaking for and in multi-agent systems is shown. The results they present are enriched by technical aspects, which are in detail considered in Section 3.1.6. Section 3.1.7 which concludes the pure matchmaking approaches by providing another very important model by Subrahmanian and coauthors. This approach is based on a very simple offer and request structure but supports powerful matching mechanisms based on Σ-hierarchies. Complexity considerations are also a central point in the authors work.

3.1.1 Application Oriented Matchmaking Mechanisms

The authors Ströbel and Stolze (2001) develop a matchmaking component for the discovery of agreement and negotiation spaces in electronic markets.

They present an *Extended Matchmaking Component* (EMC) which is designed to identify negotiable agreements in the case where genuine constraint matching does not provide successful matches. Another scenario where it is applied to is when buyers and/or sellers are willing to relax some of their constraints.

Technically the EMC is based on sets of constraints which are formulated on the buyers and sellers side. The constraints are matched using an extended matchmaking operation. This operation takes the notion of negotiable constraints into account. They state that an agreement value range exists, if two constraint instances of a seller and a buyer, respectively, propose a distance so that an agreement space is left. E.g., a buyer defines the constraint $price < \$2000$ and the seller defines $price > \$1800$. The resulting agreement value range would be $\$1800 < price < \$2000$.

If no constraints or only constraints from one side – seller or buyer – are defined, null-properties are the consequence. The authors suggest several alternatives to resolve these null-properties:

- Avoid situation
- Domain constraints are required by the participants
- Null-properties are marked as open issues and only properties with domain constraints are regarded

- Default properties are initialized which may include best practices or common trade standards

The matchmaking process generally aims on the identification of agreement candidates. These are candidates which are most likely the counterparts for a transaction execution based on a mutual beneficial deal. But the matchmaking process does not substitute the negotiation process. It is a one-shot process to determine potential counterparts for negotiations in situations where many potential negotiation counterparts are available.

3.1.2 Matchmaking in the SHADE and COINS Approach

The understanding of matchmaking underlying the approach of Kuokka and Harada (1995) and Kuokka and Harada (1996) (p. 263) and following is reflected in the following condensed definition.

Matchmaking: Matchmaking is an approach based on emerging information integration technologies where potential providers and demanders of information or goods send messages describing their information (or goods) capabilities and needs. These descriptions are taken (represented in rich machine-interpretable description languages), are unified by the matchmaker to identify potential matches and are presented to the user.

The authors of this work claim that the main purpose of matchmaking are the domains of electronic commerce and engineering.

SHADE – SHAred DEpendency Engineering The SHADE matchmaker (see Kuokka and Harada 1996) is a matchmaker which operates over logic-based and structured text languages. The aim is to dynamically connect information sources. The matchmaking process is based on KQML (see Finin et al. 1994) communication. Content Languages of SHADE are a subset of KIF (see Genesereth and Fikes 1992) as well as a structured logic representation called MAX.

KIF is a form of full first order predicate calculus. MAX is a conjunctive predicate logic augmented to support frames, which are conjunctive sets of literals, as well as strings as first class terms. The actual matching of advertised and subscribed content fields is performed by a prolog-like unification algorithm. If logic formed strings are present, a regular expression pattern matcher is used for term unification.

Matchmaking is realized solely by matching the content of advertisements and requests. There is no knowledge base and no inference performed. The SHADE matchmaker is implemented entirely as a forward-chaining rule-based program also using MAX, defined in Kuokka (1990). This allows to add features dynamically as new rules.

Advertisements and requests can contain more than a set of ground literals. In addition the SHADE matchmaker supports meta-level queries about its operation. These queries allow other agents to subscribe to the message level actions of the matchmaker in addition to content level information.
An important precondition of matchmaking is a shared ontology. Shared ontologies are needed to ensure that terms have clear and consistent semantics. Otherwise, a match may be found or missed based on an incorrect interpretation of the requests.

COINS – COmmon INterest Seeker COINS (see Kuokka and Harada 1996) is a matchmaker which operates over free text. The motivation for the COINS matchmaker is the serious need for matchmaking over the huge amount of unstructured text on the WWW and other WANs. A reason for this is that it is impossible in practice to express all these texts in KIF. Instead one has to use traditional matchmakers for an implementation on free text.

Initially the free text matchmaker was implemented as the central part of the COINS System, which was designed to allow users to easily advertise and request information on their interests. It turned out that this matchmaker is also useful as a general purpose facilitator.

Like in SHADE, access language is also KQML. The COINS matchmaker does not use the KQML brokering performatives "advertise" and "recruit" but uses the content-based routing paradigm where clients send "tells", "asks" and "subscribes". It handles two content languages: Free text and document vectors. Document vectors are weighted lists of stemmed words which occur in a document. The main advantage of document vectors is the space efficiency. The COINS matchmaker immediately converts any incoming free text into document vectors. Hence, document vectors and free text are isomorphic from the matchmaker's point of view.

With the subscribe command a client subscribes a document (-vector) to the matchmaker. As soon as a tell command comes in with a matching document (-vector) the match will be forwarded to the subscriber. Instead of whole documents the COINS matchmaker saves only the document vector which contains the name of the document, the owner as an email address. For that reason the matchmaker forwards only the document vector and the strength of the match to the subscriber. In order to obtain the whole document the subscriber has to contact the tell-agent directly.

To process and match free text and document vectors the *System for the Mechanical Analysis and Retrieval of Text* (SMART) (see Salton 1989) information retrieval system is used. The first input is free text. This text is converted into a document vector using the SMART's stemming and noise word removal. Then the document vectors are compared using an inverse document frequency algorithm.

For this purpose the frequency of a concept in a document is divided by the frequency in the entire corpus to determine its information content – i.e. how well the concept describes the document with respect to the corpus. An adjustable cutoff measure is used to make the comparison binary.

For the purpose of determining the information content of words the COINS matchmaker collects a corpus. A corpus is a database which consists of words which appeared in former queries. The semantics of the words is implicitly given by the selection of the corpus. When the matchmaker receives any new advertisement, it revises its corpus to obtain better estimates of the future computed information content of words. In practice it is the COINS matchmaker's attempt to incrementally learn the ontology already shared by its clients.

3.1.3 Service Classification in Agents' Societies

The authors Weinstein and Birmingham (1997) define service classification as a task that helps agents to select services in large and dynamic agent societies. This approach uses a Service Classification Agent (SCA) that is capable of:

- Dynamic ontology maintenance
- Usage of description logic
- Runtime organization of defined services
- Constructing taxonomy of services
- Usage of a dynamic ontology from which concepts can be used to define additional services
- Subsumption taxonomy using description logic (i.e. application of description logic to insert descriptions into a hierarchy)

In this approach the SCA takes over the role of the matchmaker/broker. An agent queries the SCA in order to find a service to complete its task. The SCA decides using the above mentioned methods which services to return to the querying agent.

The authors see an organic society of agents as a large, dynamic, evolving and terminologically heterogeneous agent community. Such a society is called proto-organic when it is a prototype for an organic society.

The subsumption taxonomy is being used to reason services with the aim to find the best service available to meet a need. The expressiveness and reasoning power of description logics may enable translation between ontologies. Ontologies are implemented because declarative descriptions of complex agent services have to be encoded. Declarative descriptions are required to establish a space of services which is independent of the implementation of the current set of agents.

Descriptions are translated manually to LOOM. For the future an automatic translation service from natural language to KIF and further to LOOM and other representative languages is integrated. The description-logic system maintains the ontology of agent services. It automatically places new concepts in taxonomies. The SCA gets a service description from a registering agent. This is classified and a label is returned, recommending to classify the service as described on the label. If the service description is new to the SCA a new concept will be introduced to the ontology. For this reason the ontologies are called dynamic.

The matchmaking process in the SCA proto-organic society approach is based on four different types of agents:

1. The **Query Planning Agent (QPA)** as an instantiation of the task-planning-agent, which is a generalized, goal pursuing procedural reasoner.
2. The **User Interface Agent (UIA)** which purchases the services. This happens using two different strategies: increasing generality (most-specific-subsuming) or increasing specificity (most-general-subsumed).
3. The **Auction Manager Agent (AMA)** defines markets and maintains an appropriate number of auctions to service each market.
4. The **Service Classification Agent (SCA)** classifies services described by a QPA and classifies requests given by UIAs.

The nested ontologies in form of a concept taxonomy are located in the knowledge base of the SCA. The idea of a meta-service classifier for large societies of agents is proposed. This meta classifier should be able to execute a society wide search for the best available service.

The authors point out that the agent system they introduce has the capability to guarantee immediate customer supply for new incoming agents. The precondition for this is that the service description matches a service request.

3.1.4 Matchmaking versus Brokering

The authors Decker et al. (1996),(p.12) focus the assumption: "The decision to use matchmaking or brokering to solve the connection problem offers many performance tradeoffs."

In order to obtain quantitative results the following measures are introduced:

1. Quantitative end-to-end response time (dis-)advantages of matchmaking versus brokering.
2. Characteristics of behaviors with respect to robust and adaptive open systems, where agents may enter and exit any time.

The system these issues are tested on is the Warren[1] – multi-agent portfolio management system. The agents communicate in KQML. The KQML performatives used are "ask", "ask-all", "stream-all", etc. Special KQML performatives like "recruit", "broker" or "recommend" are left out.

To avoid misunderstandings certain assumptions are made in order to obtain noise free comparison:

1. Ontological mismatches will not be left out.
2. Agents are sincere in their communications to one another.

Matchmaking allows one agent with some objective to learn the name of another agent that could take on that objective. Therefore, three different agent roles do exist:

1. Requester: An agent with an objective that it wants to be achieved by some other agent.
2. Matchmaker: An agent that knows the names of many agents and their corresponding capabilities. (Is often performed by a small group of agents)
3. Server: An agent that has committed itself to fulfilling objectives on behalf of another agent.

An important point is the internal commitment of the requester towards its objective: e.g. a requester agent asks a matchmaker/broker whether there is an agent offering the service to repaint my car. This does not imply that I will let my car be repainted.

So the requester-agent R believes server S cannot achieve its goal or R believes the goal might be better achieved elsewhere quality or price wise. As an important assumption, leaving out the possibility of these realistic internal commitments an agent can make towards its objective to significantly simplify the matchmaking reasoning.

Under these assumptions the role behaviors of the requester are defined as follows:

1. Find the names of agents who are willing to take on the objective.
2. Choose a server and send the initial request.
3. Form an expectation.

Requester can cache matchmaker-knowledge locally to make the matchmaking-system more robust and to reduce the load on the matchmaker. These advantages of local caching have to be faced with the disadvantage that cached data might be outdated.

Information agent: Information agents are agents that take over the tasks of answering one-shot and periodic queries, monitoring for conditions on external information sources. Information agents can handle queries stated so

[1] http://www.ri.cmu.edu/projects/project_77.html (07/01/2002)

that the requester is kept informed if the results of this query change – i.e. new agents enter the system or agents leave the system.

The matchmaking agent is an information agent. Querying the matchmaker the following properties are respected. The decision-making responsibility at a matchmade system is kept on the requesters side, on the contrary, in the broked system the broker agent takes a lot of the decision responsibility. For that reason the matchmaker should not limit the potential names returned in any way unrelated to the query itself.

Now the topic of the robustness is focussed. The basic problem is the classification of the agents' reliability. This leads to the issue of exception handling. How should e.g. an agent be treated that does not respond to a tell. If an agent does not respond, the "tell" or "ask" performative will be sent again. This process is called "murmuring". To provide stable communication, estimation of timings for answers must be specified. Active outstanding requests must be withdrawn when a commitment is abandoned to avoid dead requests. For this issue the analogous performatives at network communication protocols in real time systems can be compared.

The server role behaviors are a central aspect for consideration. Server agents must provide a set of standard behaviors. This is clearly declaring its intention by making a long term commitment called advertisement to taking on a well-defined class of future requests. This is communicated to the matchmaker using the KQML "advertise" performative. It is possible to express that some request constraints cannot be determined in advance of a specific request – e.g. price negotiations. These may be negotiated on a case-by-case basis. The authors make the assumption that all characteristics of services are fixed beforehand. If an advertisement can not be placed the well known matchmaker will be continued to attempt to advertise until the placing is successful. An agent that has taken a server role must first send a successful "unadvertise" KQML message before it can intentionally terminate or leave the system.

Furthermore the requester role behavior is defined as follows. The most important feature is that the matchmaker has a well-known name. The matchmaker participates in at least two different types of conversations: Server communication and requester communication. In this implementation matchmaker agents are a subclass of information agents and use all of their behaviors.

With growing matchmaking desire there should be more than one matchmaker in a system. Instead of putting several matchmakers next to each other there should be a complex multi-matchmaker organization which allows brokering matchmakers. Brokering matchmakers are matchmakers that may transparently contact others in order to answer queries. A multi-matchmaker can be easily and transparently be integrated into a system designed for a simple matchmaker by exchanging the simple matchmaker with a multi-matchmaker of the same name.

Brokering is different to matchmaking. At brokering one agent with an objective looks for another agent that can achieve that objective. So the broker may for a part take as well the role of the server, the matchmaker as the requester.

3.1.5 Middle Agents – Role Definitions for Mediating Agents

Due to the definition of matchmaking which is given by Decker et al. (1997) the role of a matchmaker is the connection of information providers and information requesters. A middle agent is an agent which cooperates only with other agents but not with users. Important topics of this work are the different middle agent models, the robustness and performance of these different models.

Basically three different middle agents are mentioned in the work of these authors:

1. Matchmakers or yellow page agents,
2. blackboard agents and
3. broker agents.

Important consideration is the standpoint of privacy. A request is defined as an instance of an agent's preferences and a reply is defined as an instance of an agent's capabilities. Advertisement is a capability specification. "Want-add" is a request.

Matchmaker: Middle-agent that keeps track of advertisements posted by provider agents. Requester agents can query the matchmaker to obtain a list of suitable providers which he can contact directly. Advantage is a fairly high performance. Disadvantage is a bad privacy policy for the provider agent.

Broker: Middle-agent that keeps track of both, requests and advertisements. This agent plugs fitting services into requests in order to fulfill the desired task. Advantage is a good privacy policy. Neither provider nor requester know each other. The disadvantage of centralized load balancing is a communication bottleneck. A broked system is critical non-robust due to the possibility of a broker failure.

Blackboard: Middle-agent that keeps track of requests. Providers can query the blackboard in order to obtain services they are capable of handling. Advantage is again the high performance. Disadvantage is the bad privacy policy for requester agents.

Hybrid constructs between these middle-agents are possible: e.g. using an anonymized middle-agent and contacting through this a matchmaker comparable to using a broker agent.

Another interesting hybrid is a combination of matchmade and broked system. Here for example a matchmaker holds the advertisements of a bunch of broker agents. A provider contacts the matchmaker to get the address of the appropriate broker and advertises with this one. This combination avoids in some cases disadvantages and provides a higher robustness.

Decentralized systems like matchmade or blackboarded systems with local caching are much more robust than centralized – e.g. brokered – systems. Of course it makes more sense to store capabilities locally instead of requests. These stay up to date for a longer time.

The authors compare the robustness of a brokered system with the robustness of a matchmade system without cache, i.e., each requester has to contact the matchmaker every time to obtain a list of services before contacting an appropriate provider. The result is that both perform equally well.

Matchmade/Brokered hybrid: The most interesting hybrid combination between matchmade and brokered systems is the following: Let the system be a brokered one as long as the broker does not fail. As soon as the broker fails it switches automatically to a matchmade system with the loss of provider's privacy.

In the role behavior the following changes have to be made. Brokers must advertise their brokering as a capability to a matchmaker, and providers must initially register with the matchmaker for broker-capability updates.

The most important conclusion of this paper is the definition of a hybrid matchmaking/brokering system. This system uses brokering as default and switches to matchmaking in case of a broker failure. As shown in the definitions of the characteristics of the middle agents, brokered system have in contrast to matchmade systems the advantage to guarantee privacy of the provider. If this hybrid system is used in case of emergency the privacy of the provider agent is being abandoned.

3.1.6 LARKS: Interoperability among Heterogeneous Software Agents

The authors Sycara et al. (2002), Sycara et al. (1998), Sycara et al. (1999), Jha et al. (1998) and Sycara (1999) follow the objective to create a matchmaking approach that connects heterogeneous agents in open environments like the Internet. To achieve this they define a language called Language for Advertisement and Request for Knowledge Sharing (LARKS).

Similar to their earlier approaches the authors of this work start with distinguishing different middle agents:

- matchmaker agents, yellow page services – see Decker et al. (1996), Decker et al. (1997), Arisha et al. (1999)

- broker agents – see Decker et al. (1996), Decker et al. (1997)
- billboards – see "blackboard" in Section 2.1.2

The process of a requester finding an appropriate provider through a middle agent they call matchmaking. The following steps describe the communications that take place.

- Provider advertises its capabilities (know-how, expertise, etc.) to a middle agent
- Middle agent stores these advertisements
- Requester asks middle agent for providers with the desired capabilities
- Middle agent matches the request against stored adds and returns result to requester
- Requester communicates with providers

The specific topic the authors of this work address is the heterogeneity and incapability of general understanding of requesters and providers in heterogeneous surroundings.

There is the need for a common language for describing the capabilities and requests of software agents in a convenient way. Then there must be developed an algorithm in which descriptions in that language can be matched. If a solution for matchmaking among heterogeneous agents is planned, the following agent roles have to be defined:

1. **Provider agents:** Provide their capabilities, e.g. information search, provision of special products of electronic commerce to other agents or a user.
2. **Requester agents:** Consume information and services offered by provider agents in the system.
3. **Matchmaker agents:** Mediate among both, for some mutually beneficial cooperation. Each provider must first register himself with a matchmaker. Providers send an appropriate message to the matchmaker describing their capabilities and kinds of services they offer. Every request received will be matched with the actual set of advertisements. If the match is successful a ranked set of appropriate provider agents is returned.

The advantage of a matchmade system towards a broked system is that bottlenecks are avoided. A disadvantage is the possibly higher amount of agent communication.

The authors formulate a set of desiderata for an Agent Capability Description Language (ACDL). These desiderata lead to the definition of LARKS.

- Expressiveness: The ACDL must be able to describe data, knowledge as well as program code. Agent capabilities are described at an abstract rather than implementation level.
- Inferences: Any pair of descriptions in this language can be read by a user and can be compared automatically.

- Ease of Use: The language must be easy to read and easy to write by any user. The language must avoid redundant work for the user and improves the readability of specifications.
- Application on the Web: Advertisements and requests of agents on the Web must be processable.

The matchmaking process must be efficient, most accurate and not only rely on keyword extraction and comparison. This process should be fully automated. An implementation of a language which supports these features is LARKS.

`Context`	Context of specification
`Types`	Declaration of used variable types
`Input`	Declaration of input variables
`Output`	Declaration of output variables
`InConstraints`	Constraints on input variables
`OutConstraints`	Constraints on output variables
`ConcDescriptions`	Ontological descriptions of used words

Table 3.1. Specification Slots in LARKS

Specifications in LARKS: A specification in LARKS is a frame with the structure shown in Table 3.1.
The frame slot types in this structure mean the following:

- `Context`: Context of the specification in the local domain of agents.
- `Types`: Declaration of the used data types.
- `Input` and `Output`: I/O variables for required in-/output knowledge to describe a capability of an agent.
- `In-/OutConstraints`: Logical constraints on in-/output variables in the in-/output declaration part. The constraints are specified as Horn clauses.
- `ConcDescriptions`: Optional description of the words used in the specification (includes domain ontologies).

Local domain ontologies are described in a structure called concept. A concept is represented in the Information Terminological Language (ITL). A generic interface to be able to understand ontologies in languages which are different from ITL is being implemented. Every LARKS specification is wrapped up into a KQML statement to be communicated to the opposite agent.

The main benefits of using concepts are as follows:

1. The user can specify in more detail what he is requesting or advertising.
2. The matchmaker agent is able to make automated inferences on such kind of additional semantic descriptions while matching LARKS specifications, thereby improving the overall quality of matching.

Among concepts there exist certain subsumption relationships. On these subsumption hierarchies the matchmaking process is partially based.

A concept C subsumes another concept C', if the extension of C' is a subset of that of C. This means that the logical constraints defined in terms of the concept C' logically imply those of the more general concept C. Any concept language is decidable if it is defined for concept subsumption among two concepts in that language. The concept language ITL which is used here is NP-complete. The computation of subsumption relationships among all concepts in an ontology yields a so-called concept hierarchy. These as well as subsumption relationships are used in the matchmaking process.

The matchmaker stores its knowledge in the following forms:

1. Advertisement database (ADB) contains all advertisements written in LARKS that the matchmaker receives from provider agents.
2. A so-called partial global ontology is collected by the matchmaker and consists of all ontological descriptions of words in advertisements stored in the ADB.
3. Auxiliary database for the matchmaker consists of word pairs and word distances, basic type hierarchy and internal data.

The five steps of Matchmaking in LARKS

1. **Context Matching:** Select those advertisements in the ADB which can be compared with the request in the same or similar context.
2. **Syntactical Matching:** This filter compares the request with any advertisement selected by the context matching in three steps:
    1. *Comparison of profiles:* Standard technique from the Information Retrieval area, called term frequency-inverse document frequency weighting (TF-IDF) Salton (1989). In this part of the matchmaker any specification in LARKS is treated as a document.
    2. *Similarity matching:* Similarity of the input/output variable declarations as well as input/output constraints is computed by a similarity algorithm and returned as a real value.
    3. *Signature matching:* Matches the variable types of the input and output variables.
3. **Semantical Matching:** This final filter checks if the input/output constraints of any pair of request and advertisement logically match. Two specifications S and T match semantically if their pre- and post-conditions Pre_S and $Post_S$ respectively match. This means the set of pre-conditions of S logically implies that of T and the set of post-conditions of S is logically implied by that of T:

$$(Pre_S \rightarrow Pre_T) \wedge (Post_T \rightarrow Post_S)$$

   Therefore the θ-subsumption with a set of Horn clauses is used. This subsumption is weaker than logical implication.

At the LARKS matchmaker four different types of matches are distinguished:

1. **Exact Match (EM):** The most accurate matches are exact matches. Both descriptions are equivalent, either equal literally or equal by renaming the variables or equal logically obtained by logical inference.
2. **Plug-In Match (PIM):** Less accurate but more useful match. This kind of match is obtained when certain matching criteria are completely fulfilled and others do not match at all, because one of the fields to be compared is left blank. Exact match is a special case of plug-in match.
3. **Relaxed Match (RM):** Least accurate but most useful. Much weaker semantical interpretation than the first two kinds of matches. It will not be returned whether two descriptions will semantically match but return a numeric distance value. Normally the plug-in match and the exact match are special cases of the relaxed match if the threshold value is not too small.
4. **Profile matching (PMM):** This matching mode only matches the profile and the context.

Table 3.2 shows which matchmaking steps of the LARKS matchmaker are used for which type of match.

	EM	PIM	RM	PMM
`1. Context Matching`	●		●	●
`2. Profile Matching`	●		●	●
`3. Similarity Matching`	●		●	
`4. Signature Matching`	●	●		
`5. Semantical Matching`	●	●		

Table 3.2. Five Steps of Matching in LARKS

The computation of semantic distances among concepts is executed. This is done because of the existence of concept subsumption and a relation based on the concepts of the profiles. This is only a generalization or specialization relation among concepts. To express additional associations among concepts, there is a weighted associative network implemented. This is a semantic network with directed edges between concepts as nodes. Any edge denotes the kind of a binary relation among two concepts and is additionally labeled with a numerical weight interpreted as a fuzzy number. This weight indicates the strength of belief in that relation since its real-world semantics may vary. The relationships are fuzzy and can not possibly associate all concepts with each other. The semantic network consists of three kinds of binary weighted relationships: 1. generalization, 2. specialization, 3. positive association among concepts which is the most general relationship among concepts in the network indicating them as synonymous in some context. Such a network is called *Associative Network* (AN).

3.1.7 Matchmaking Approaches for Heterogeneous Active Agents

The following matchmaking approach is introduced in technical and formal detail. This is done due to the fact that key aspects of this approach motivated the definition of the GRAPPA formalisms.

Subrahmanian et al. (2000) and Arisha et al. (1999) present a complete open multi-agent society based on matchmaking via a yellow pages server. The system is called Interactive Maryland Platform for Agents Collaborating Together (IMPACT). First of all the system is described briefly. The IMPACT system consists of the following components:

1. **Provider agent** is an agent that provides its capabilities (described in services) to other agents.
2. **Requester agent** is an agent that seeks for an agent that performs the desired task or offers the information which is looked for.
3. **Yellow-Pages-Server** is the centralized matchmaker which connects providers and requesters.

The YP-server uses certain auxiliary servers to process the matchmaking. These auxiliary servers are the *Type Server* which collects the type hierarchies and the *Registration Server* which is always contacted when a new agent registers at the YP-server. This server inserts new types into the type hierarchy of the type server as well as registers new services in the relational agent database of the YP-server.

The YP-server uses two different matchmaking approaches which can be requested by each agent. The first method is the *k-nearest-neighbor-algorithm*. This algorithm returns the closest k services that match the desired service. The second method is the *range-algorithm*. This method returns all services that are within the specified range.

For the realization of both algorithms distance functions are needed. To define these distance functions a metric is needed to compute the distance of two service descriptions.

Service description: A **service name** is a special type called VerbNounterm:

$$v : n_1(n_2)$$

Where v is any verb in English language, n_1 and n_2 are any noun in English language.

The verb and noun are inserted into a verb- and a noun-hierarchy. These hierarchies are directed acyclic graphs which have weighted edges. The metric defined on these graphs is the sum of the weights on the shortest way from one node to the other. If there is no such way, and the graph-parts in which the nodes are located are disjoint, the distance is ∞. If the two nouns or verbs which are compared are in the same node, the distance is 0. These ideas are formalized in the following definitions: Let Σ be a set of English verbs or NounTerms.

Σ-node: A Σ-node is a subset $N \subseteq \Sigma$ that is closed under a relation $\sim$, which means

1. $x \in N \wedge y \in \Sigma \wedge y \sim x \Rightarrow y \in N$
2. $x, y \in N \Rightarrow x \sim y$.

So Σ-nodes are equivalence classes of Σ.

Σ-hierarchy: A **Σ-hierarchy** is a weighted, directed acyclic graph $\mathcal{SH} := (T, E, \rho)$, where E is the set of edges, such that:

1. T is a set of nonempty Σ-nodes.
2. For t_1 and t_2 are different Σ-nodes in T, then t_1 and t_2 are disjoint.
3. ρ is a mapping from $\rho : E \to \mathbb{N}$ a positive distance between two neighboring vertices.

Distance: Given a Σ-hierarchy $\mathcal{SH} := (T, E, \rho)$, the distance between two terms, $w_1, w_2 \in T$, is defined as follows:

$$d_{\mathcal{SH}}(w_1, w_2) := \begin{cases} 0 & \text{if some } t \in T \text{ exists such that } w_1, w_2 \in t. \\ \text{cost}(p_{\min}) & \text{if there is an undirected path in } \mathcal{SH} \text{ between} \\ & w_1 \text{ and } w_2 \text{ and } p_{\min} \text{ is the path generating} \\ & \text{least cost.} \\ \infty & \text{otherwise.} \end{cases}$$

It is easy to see that given any Σ-hierarchy the distance function $d_{\mathcal{SH}}$ induced by it is well defined and satisfies the triangle inequality.

Composite Distance Function: Suppose we have two different sets of words Σ_1 and Σ_2 with Σ_1-hierarchy $\mathcal{SH}_1 := (T_1, E_1, \rho_1)$ and Σ_2-hierarchy $\mathcal{SH}_2 := (T_2, E_2, \rho_2)$. Let d_1 and d_2 be the distance functions induced by $\mathcal{SH}_1$ and $\mathcal{SH}_2$, respectively. Consider three pairs of words $\langle w_1, w'_1 \rangle, \langle w_2, w'_2 \rangle, \langle w_3, w'_3 \rangle \in \Sigma_1 \times \Sigma_2$. A composite distance function cd is any mapping from $(\Sigma_1 \times \Sigma_2) \times (\Sigma_1 \times \Sigma_2)$ to $\mathbb{N}$ such that

1. (Symmetry) $cd(\langle w_1, w'_1 \rangle, \langle w_2, w'_2 \rangle) = cd(\langle w_2, w'_2 \rangle, \langle w_1, w'_1 \rangle)$.
2. (Ipso-distance) $cd(\langle w_1, w'_1 \rangle, \langle w_1, w'_1 \rangle) = 0$.
3. (Expansion of d_1) If $d_1(w_1, w_2) \leq d_1(w_1, w_3)$, then
$$cd(\langle w_1, w'_1 \rangle, \langle w_2, w'_2 \rangle) \leq cd(\langle w_1, w'_1 \rangle, \langle w_3, w'_2 \rangle).$$
4. (Expansion of d_2) If $d_2(w'_1, w'_2) \leq d_2(w'_1, w'_3)$, then
$$cd(\langle w_1, w'_1 \rangle, \langle w_2, w'_2 \rangle) \leq cd(\langle w_1, w'_1 \rangle, \langle w_2, w'_3 \rangle)$$

5. (Triangle inequality)

$$cd(\langle w_1, w_1' \rangle, \langle w_3, w_3' \rangle) \leq cd(\langle w_1, w_1' \rangle, \langle w_2, w_2' \rangle) + cd(\langle w_2, w_2' \rangle, \langle w_3, w_3' \rangle).$$

A simple composite distance function is the sum, the addition of two values. Using these definitions the matchmaking algorithms k-nearest-neighbor and range computation can be introduced.

***k*-Nearest-Neighbor Algorithm** The k-nearest-neighbor algorithm is an algorithm that returns the k services which are closest to the described service. The variables and functions used in the algorithm shown in Table 3.3 define as follows:

1. **Todo:** List of $\langle v : nt \rangle$, extended by distances from initially or recursively (in the find-nn function) requested $\langle v : nt \rangle$ pair. List is maintained in increasing order of distance, not necessarily complete.
2. **ANSTABLE**: List of the closest k neighbors to the requested $\langle v : nt \rangle$ pair found thus far. Maintained during computation. Consists of $\langle v : nt \rangle$ pairs and distance to the requested term in increasing order.
3. **search-service-table:** This function returns a set of all k agents which provide the service $\langle v : nt \rangle$. If the number exceeds k, k of them are deliberatively chosen (e.g. randomized).
4. **num-ans:** Counts the number of answers kept in ANSTABLE.
5. **next-nbr:** This function takes as input the Todo-list. This list consists of $\langle v : nt \rangle$ pairs that have not yet been relaxed.
   Suppose $\langle v_1 : nt_1 \rangle$ is the first pair in the Todo-list. A *candidate-relaxation* of $\langle v_1 : nt_1 \rangle$ is a relaxed pair $\langle v' : nt_1 \rangle$ where v' is an immediate neighbor to v_1 or a pair $\langle v_1 : nt' \rangle$ where nt' is an immediate neighbor of nt_1. The function next-nbr removes $\langle v_1 : nt_1 \rangle$ from the Todo-list and inserts all candidate-relaxations of $\langle v_1 : nt_1 \rangle$ with the distance to the original pair instead. As an output the first member of the new Todo-list is returned.
6. **relax-thesaurus:** This function is called when either v or t of the pair $\langle v : nt \rangle$ does not appear in the verb- or noun-term-hierarchy. It returns a pair that is similar to $\langle v : nt \rangle$, whose components appear in the hierarchies.

Range Algorithm The range-algorithm is the second neighbor finding possibility in the IMPACT system. This algorithm takes a distance n as input and returns all neighbors within this distance. It is possible that no services are close enough so that the returned set is the empty set.

The variables and functions used in the algorithm shown in Table 3.4, additionally to the ones which have already been introduced for k-nearest-neighbor, define as follows:

```
create(Todo,V,NT);
ClosedList:=NIL;
ANSTABLE:=0;
**if** ⟨V,NT⟩ $\in \Sigma_v \times \Sigma_{nt}$ **then**
{
    done:= false;
    Sol := search-service-table(V,NT,K);
    **while** ¬done **do**
    {
        insert(⟨V,NT⟩,ClosedList);
        insert(Sol, ANSTABLE);
        n := num-ans(ANSTABLE);
        **if** $n \geq K$ **then** done := true
        **else**
        {
            ⟨V',NT'⟩ := next-nbr(Todo);
            **if** error(V',NT') = true **then** done := true
            **else**
            {
                ⟨V,NT⟩ := ⟨V',NT'⟩;
                Sol := search-service-table(V,NT,K-n);
            }
        }
    }
}
**else**
{
    (* search thesaurus *)
    ⟨V',NT'⟩ := relax-thesaurus(V,NT);
    **if** error(V',NT') = true **then** return ERROR
    **else** find-nn(V',NT',K)
}
return ANSTABLE;
end.
```

Table 3.3. *k*-nearest-neighbor Algorithm

1. The **Todo-List** is defined like in the *k*-nearest neighbor algorithm.
2. **Step I**: The first step is the while loop. All pairs $v^* : nt^* = \langle V^*, NT^* \rangle$ which are within the distance D of $v : nt$. The obtained elements are inserted into the Todo list.
3. **Step II**: A select operation over the service table is executed, finding all agents that offer any of the service names identified in the first step. Like in *k*-nearest-neighbor the method relax-thesaurus is called to find a similar pair which belongs into the list of returned statements.

```
if D<0 then return NIL(exit);
if ⟨V,NT⟩ ∈ $\Sigma_v \times \Sigma_{nt}$ then
{
    RelaxList:=NIL;
    Todo := ⟨⟨V,NT⟩, 0⟩;
    while Todo≠NIL do
    {
        ⟨⟨V',NT'⟩,D'⟩:= first element of Todo;
        insert ⟨⟨V',NT'⟩,D'⟩ into RelaxList;
        remove ⟨⟨V',NT'⟩,D'⟩ from Todo;
        expand (⟨V,NT⟩,⟨V',NT'⟩,D);
    }
    return $\pi_{\mathrm{Agents,Dist}}$(RelaxList[Verb=V',NounTerm=NT']ServiceTable)
}
else
{
    (*search thesaurus*)
    ⟨V',NT'⟩:=relax-thesaurus(V,NT);
    if error(V',NT') then return ERROR
    else return range(V',NT',D-cd(⟨V,NT⟩,⟨V',NT'⟩))
}
end.
```

Table 3.4. Range-algorithm

3.1.8 The InfoSleuth Approach

InfoSleuth (see Nodine et al. 1999) is a system for the discovery and retrieval of information in open and dynamically changing environments. The included brokering function provides reasoning over both – the advertised syntax and semantics. The purpose of the InfoSleuth system is the cooperation among several software-agents for information discovery. The agents can take three roles:

- Core agents: Broker agent, Task planning agent, Multiresource agent, Data mining agent, Ontology agent.
- Resource agents: Agent wrapping resources.
- User agents: Agent representing users.

One central service for the participating agents is the broker agent's service. The broker agent is equipped with a matchmaking service which is matching agents that require services from other agents that can provide those services. To apply this procedure, an advertising agent has to register with the broker agent. The broker inserts the agents description into its broker repository. For the description of a service only vocabulary is used which is present in the brokers service ontology.

The broker can then execute queries by requesting agents. These queries are formulated by agents which need other agents to fulfil their tasks. The broker uses a rule-based reasoning engine implemented in LDL (see also Zaniolo 1991). This engine is used to reason over the query and advertisements, to determine which agents have advertised services that match those requested in the query.

The system also uses CORBA and KQML for syntactic communication issues. As content languages SQL and LDL are allowed.

Special attention has to be payed to the multibroker architecture which has been realized within the InfoSleuth project. In this multibroker environment a broker does not keep its information private but shares it with other agents. Brokers, active in the system, are connected in the structure of a directed graph. The nodes are representing brokers and the arcs are representing knowledge of other brokers current advertisements. If broker A has advertised itself to broker B, then there is an arc from A to B. The connectivity within the system is sufficient so that each broker is either directly or indirectly connected to each other broker in the system. A broker consortium is a set of brokers which are fully interconnected. The application of broker consortia do enlarge the range of brokers and enhance the performance of a broker system.

3.2 Electronic Negotiation Approaches

Negotiation in the classical sense is a wide field. According to the Oxford Advanced Learner's Dictionary a negotiation is a "discussion aimed at reaching an agreement". This very general understanding has to be clustered into different domains in order to delimit properties for the formulation of potent electronic negotiations. Ströbel and Weinhardt noted that "unfortunately, there does not exist a comprehensive taxonomy of traditional negotiations, which could be used as a foundation for the development of an electronic negotiations taxonomy" (Ströbel and Weinhardt 2002, p.2). For this reason it is hard to delimit negotiation domains in which methods and procedures can be formalized to electronize negotiation processes. However, for the purpose of this work negotiations in the economic sense are addressed. Concrete approaches for these negotiations are outlined, ideas for electronic procedures are provided and negotiation engines and negotiation support systems are introduced.

In this section many approaches from different research fields are outlined briefly. Most of these research projects are interdisciplinary and include research from different disciplines. To still provide a topical order, they are presented in the following sequence:

- Different contributions form the same author/research group are clustered
- Different contributions treating similar problems are clustered

Of course these two aspects do sometimes clash. For that reason author/group associated contributions are sometimes separated.

In the next section a part of the "Montreal Taxonomy for Electronic Negotiations" by Ströbel and Weinhardt (2002) is presented. This part describes a framework which is used to classify negotiation research approaches using homogenous criteria.

3.2.1 Electronic Negotiation – The Montreal Taxonomy

The model to structure electronic negotiations which is presented in Ströbel and Weinhardt (2002) allows to separate several phases within an electronic negotiation and enables the integration of matchmaking into electronic negotiation scenarios. It also emphasizes the key role matchmaking is playing in such negotiation scenarios.

The taxonomy is based on the general notion and understanding of negotiations: A negotiation is an iterative communication and decision making process between two or more sides (parties) represented by two or more agents who cannot achieve their objectives through unilateral actions and who search for a consensus decision. Additionally trade negotiations are considered, i.e. the negotiation processes in electronic markets for the exchange of goods and services based on bargaining, bidding, or dispute resolution, and do not take into account non-commercial domains or other forms of negotiations such as group decision-making or voting.

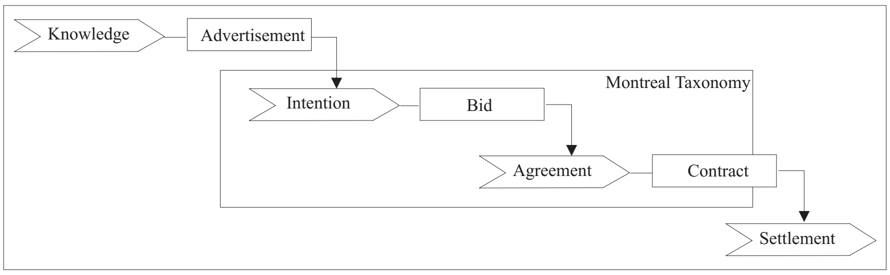


Figure 3.1. Interaction Phases of Electronic Negotiations

In Figure 3.1 the interaction phases are emphasized which are focussed in the Montreal Taxonomy. The authors state that a negotiation process takes place when the first agreement fails – i.e. when based on the offers made in the intention phase, either an agreement cannot be reached or the agreement has potential for optimization.

The authors focus on the process execution tasks within a negotiation. This has already been complemented in Section 2.3. The authors distinguish between

- Explicit criteria
- Implicit criteria

As well as

- Exogenous criteria
- Endogenous criteria

The combination of both categorization approaches results in four criteria categories. Within the explicit criteria distinctions in the business domain as well as the business model are made. The business domain criteria should cover the relation of the agents operating in the business domain (e.g. the buyers and the sellers). Whereas the business model defines the role of a business (e.g. electronic negotiation provider) within the described business domain.

Also within the implicit criteria this distinction is made. The business domain here is focusing on exogenous implicit criteria which might describe the culture of the market community (e.g. from a trust perspective). The business model comprises implicit criteria on a high level such as the mission with a value proposition.

Within the endogenous classification criteria the explicit criteria are summarized as the following:

1. Roles: Participation, agents, admission, identity, collusion.
2. Process – Overall rules: Variation, rounds, stages, concurrency.
3. Process – Offer specification: Attributes, values, relaxation, structure, relation, object.
4. Process – Offer submission: Sides, position, activity, direction.
5. Process – Offer analysis: Value, threshold.
6. Process – Offer matching: Schedule, sorting, evaluation, resolution.
7. Process – Offer allocation: Distribution, provision, configuration,
8. Process – Offer acceptance: Commitment.
9. Information: Communication, transaction, negotiation, transparency, trace, content, timing.
10. Strategy: Fees, arbitration, ratings.

Within the offer matching process (6.) the subprocedures are addressed as follows:

a) Schedule
   i) Offer matching takes place in defined time intervals (clocked).
   ii) Offer matching is triggered by events (e.g. the submission of an offer).
   iii) Permanent execution of offer matching (continuous).

b) Sorting
   i) For a satisfying sorting agents can define in their offers constraints towards the potential transaction partners, which have to be evaluated for the offers to match. Depending on the type of match (full compliance, negotiable violations, etc.) matching pairs of offers are assigned to classes without further ranking.
   ii) The process sorting is broadcast if no difference is made among the potential transaction partners – all are qualified in the same way for the consecutive process.
   iii) In the case where potential transaction partners are a priori chosen by one or several agents, the sorting is exclusive.

c) Evaluation
   i) In a ranking scenario, offers are ordered according to the preferences of an agent in order to find out the "best" offer among the set of matching offers. Ranking can be applied to the results of a preceding classification.
   ii) If no offer scoring is executed, the electronic negotiation medium is just listing the matching offers (which still might be classified).

d) Resolution
   i) The process has a defined resolution if a rule-driven selection (e.g. "one offer from each seller or class" or "only three offers") is executed for the set of classified or scored offers and only suitable offers are considered for the consecutive process.
   ii) A defined resolution may be complemented with a tie-breaking rule, which defines how selection conflicts are resolved. Conflicts arise if, for instance, there are multiple candidates for the execution of the transaction found through sorting and/or scoring, but only one can be chosen.
   iii) If no resolution or tie-breaking rules are defined, conflicts are forwarded and have to be resolved by the agents.

This shortened citation from Ströbel and Weinhardt (2002) shows that the matching phase is a central procedure within electronic negotiations. Especially in the phase of c) "Evaluation" i), there are numerous algorithms in theory and practice which can be applied to solve the matchmaking problem between offering and requesting agents. Depending on the algorithm chosen, even the negotiation rules might vary strongly.

Besides the taxonomic approach to electronic negotiations the authors made several contributions to the field: In Ströbel (2000a), the author focuses on the relation of the implication of specific electronic market characteristics to the effectiveness of major types of negotiations. The analysis reveals why bidding protocols currently dominate bargaining protocols and suggests that future negotiation support beyond auctions should be based on integrative multilateral protocols. In Ströbel (2000b) the author introduces a procedural framework for an electronic market mediation service based on the adjusted-winner procedure for fair distribution. In Ströbel (2001a) a communication scheme for electronic negotiations is introduced, which is based on XML schema. This schema explicitly specifies the common syntax and semantics of the negotiating parties, the logical space of the electronic negotiation. Finally, Ströbel (2001b) demonstrates how an explicit and specific design can capture the way electronic negotiations are organised. The design meta-model presented is part of SILKROAD, a design and application framework for electronic negotiations developed by the author.

3.2.2 Electronic Negotiation Approaches with Information Systems Focus

In this section some of the current electronic negotiation approaches are outlined. The research directions, which are at the current state of knowledge most important for this work are described.

Bichler (2001) and Bichler et al. (1998a) regard electronic brokerage as the core concept necessary to overcome the current limitations of Internet commerce and to fulfill the purpose of traditional intermediaries. Bichler states that "currently most electronic brokers on the Internet concentrate on aggregation of information from underlying electronic catalogs". Due to the fact that currently information aggregation is static, i.e. the so called broker gathers information before a user requests it, today's brokers are equivalent to Internet marketplaces or mediating agents.

Electronic brokers provide a trusted conjunction of supply and demand processes. Brokers are defined alternatively compared to the broker definition in the multi-agent domain, introduced in Section 2.1.2. The key property, which defines a broker as a specific party mediating between supply and demand, is the fact that it provides anonymity for the provider and the requester. Furthermore a broker – understood as a mediating party – has the following core services supporting the matching of consumer interests and supplier abilities (Bichler 2001, p.29):

- "Mechanisms through which a supplier can express an ability to provide products and/or services and though which a consumer can express an interest in products and/or services."
- "Mechanisms to dynamically aggregate information from registered electronic catalogs, i.e. to enable users to get the appropriate perspective on the supplier domain."
- "Mechanisms supporting notification capabilities, i.e., a supplier can supply a description of abilities to a broker for the purpose of propagating the ability to registered consumers who have expressed an interest in the particular ability or vice versa."

- "Mechanisms for bilateral and multilateral negotiations about products and services. Negotiation support can be considered the most complex form of matching buyers and suppliers in a marketplace."

A broker also needs to offer various services, depending on the kind of product. As in physical markets, there must be brokers for different application domains e.g. real estate, cars, human resources, etc. who need to provide different functionalities to match supplying and demanding parties. Bichler depicts the architectural model for electronic brokerage as a framework combining from mainly four aspects:

- the yellow pages
- the catalog aggregation
- the notification service, and
- the negotiation service

Yellow pages services mediates long-term market information about the market participants. The catalog aggregation describes the task of consolidating information from a selection of suppliers catalogs. The notification service pushes time-sensitive market information such as restricted offers or requests for bids to market participants. Finally the negotiation service, the most challenging of these four aspects, describes the task of dynamically negotiating the negotiable attributes of a product or service. The design of the negotiation service has to respond to a broad set of questions (Beam et al. 1999) concerning

- technologies
- rules of the marketplace, and
- strategies for the participants

In Bichler et al. (1998b), Bichler and coauthors describe a negotiation service called *a broker-based Object Framework For Electronic Requisitioning* (OFFER) which provides an electronic broker instance. Within this electronic broker service, the buyer-seller negotiation session was replaced with a set of auction mechanisms. The strategy issue for bidders is collapsed into a single dimension of bid formulation. The OFFER service implements a range of different auction mechanisms: an English, Vickrey, and first-price sealed-bid auction. In general, the question which negotiation protocol (out of a set of protocols) is best is non-trivial. The auction protocols and a basic comparison of these is given in Section 3.2.4.

The difference between negotiation and optimization questions is the consideration of private information. Also, often in real life negotiation situations there is an incentive to misrepresent the own preferences (see also Oliver 1996a). Although both sides are in a competitive situation they are looking for an agreement with a mutually optimized outcome.

A basic principle in microEconomics and negotiation sciences is that there is no single best protocol for negotiations which is suitable for all negotiation situations. A general demand on a negotiation protocol which is considered for a specific negotiation situation is, that the protocol is optimal with respect to reach the best negotiation outcome for both participating parties. In most current real life situations the applied negotiation protocols stay far behind this optimum.

Since different negotiation protocols are appropriate in various negotiation situations, an electronic broker or mediator should offer a set of different negotiation protocols (see Budimir et al. 2002, Wurman et al. 1998, Wurman et al. 2001 and Wurman 1999).

In Bichler (2001), p. 82 , the state-of-the-art in automated negotiation practices are summarized. In the view of economists it is important to distinguish homogenous and heterogeneous products and services. While the former are easily traded in automated negotiations, the latter are hard to formalize and hence are hard to trade on electronic platforms. Most approaches to electronic negotiations made today are designed for homogenous items. Auction mechanisms such as the ones introduced in Section 3.2.4 define easy-to-implement electronic negotiation mechanisms for such items. Furthermore one-on-one bargaining situations (1:1) and multiple participant negotiations ($n : m$) are considered. Also a distinction is made between situations in which only a single unit of a product is traded at a certain point in time or multi-unit negotiations are performed.

Beam and Segev (1997) also gave an overview on state-of-the-art automated negotiations. They define negotiation in electronic commerce as the process by which two or more parties multilaterally bargain resources for mutual intended gain, using the tools and techniques of electronic commerce. A distinction is made between "open" and "closed" marketplaces. Closed marketplaces are based upon a predefined set of users who "enroll" in the marketplace and agree to a certain set of rules. An open marketplace has no such agreement. Any agents are welcome to enter and exit at any time, and are required to agree to no rules. Due to the authors the following difficulties appear when automating negotiations:

- The need for a common ontology is given. Concerning this issue the authors Mädche et al. (2002) and Staab (2002) gave significant approaches on automated onotology learning approaches using the Semantic Web.
- The need for a strategy for the participating agents.
- The capability of agents to be able to evaluate the tradeoffs and implication of all variables, i.e. attributes, which are negotiable.
- The privateness of the agents' negotiation strategy information: If one's agent's negotiation strategy is known to the other agent, the first agent may be at a significant disadvantage.

The authors also focus on Negotiation Support Systems (NSS), which they see as an implementation paradigm which is specially geared towards helping human negotiators make better decisions and negotiate more productively. A prominent example for a NSS is the Carleton University's INSPIRE[2] system, see also Lo and Kersten (1999). INSPIRE is an interactive system which helps two human users negotiate a solution to a predetermined problem. In the INSPIRE system also experiments with cross-cultural anonymous negotiations have been made. The system features the verification of negotiation efficiency. This has been evaluated and several reasons for negotiators to accept inefficient compromises have been proposed in Kersten and Cray (1996). As the authors Madanmohan et al. (2000) show, the INSPIRE system is also used to train negotiation skills of human users.

Bichler et al. (2002) focus on the engineering of negotiations. This comprehensive contribution focuses firstly the research areas which influence current eNegotiation research. The authors state that the impact of information technologies (IT) on negotiations is not limited to the use of electronic communication in addition to face-to-face communication. IT changes ways the negotiation problem can be represented and the ways the negotiation process can be structured. The authors provide a large variety of research fields, influencing negotiations. The main contributions include studies of law, social sciences as well as economic sciences and computer science. Computer science and information systems contributions started by designing decision and negotiation support systems and went on automating negotiations using artificial negotiating agents, software platforms for bidding and auctions as well as electronic negotiation tables.

However, the authors contribute that Internet based information systems and other systems deployed on the Internet provide efficient matching capabilities for potential negotiators, efficient exchange of information as well as the tools for data collection, data mining, analysis and interpretation.

Especially multi-attribute auction and negotiation protocols which have all the features of many complex negotiation situations are likely to gain importance. In the area of multi-attribute auctions bids on several attributes are allowed (see Bichler 2001 and Bichler 2000a). Protocols allowing bilateral negotiations in several attributes are introduced by Kersten (1999) and Ströbel (2000b).

The computerization of processes in negotiations affects the way companies interact with their customers, suppliers and business partners. Since firms carried out negotiations with a counterpart traditionally in bilateral manner negotiations used to be very time consuming and intransparent. The process is not efficient and suffers from limited transparency of negotiated issues, e.g. , the price. However, the promise of electronic negotiations is a higher level of

[2] http://interneg.org/inspire (06/25/2002)

efficiency and effectiveness, better quality of negotiation outcomes as well as faster emergence of solutions. The exchange of information will quantitatively and qualitatively improve during the negotiation process.

Klemperer (1999) sees the term negotiation as a synonym for bilateral bargaining where the understanding of other scientists like Raiffa (1996) have a much broader view. On the other hand in computer science related literature negotiations are considered as almost any message exchange between independent computer systems, or – in electronic business applications – any structured message exchange used within a negotiation such as an auction. This view is also promoted by Wurman et al. (2001).

In the domain of information management and systems the authors Bichler et al. (2002) (p.5) understand negotiation as:

1. An iterative communication and decision making process between two or more agents (parties or their representatives),
2. who exchange information comprising offers, counter-offers and arguments,
3. whose tasks are independent,
4. who cannot achieve their objectives through unilateral auctions,
5. who search for a consensus which is a compromise decision.

They define the outcome of a negotiation as a compromise which is an agreement accepted by all involved participants or a disagreement.

A negotiation is called "unstructured" when no rules are specified a priori (see also Ströbel and Weinhardt 2002). The aim in defining negotiation protocols is to:

1. Restrict the freedom of the negotiator and lead them to compromises.
2. Increase the efficiency of the process.
3. Make sure that better outcomes are achieved (e.g. at least Pareto efficiency)

Based on their view on "negotiation" Bichler et al. (2002) now see an electronic negotiation as a negotiation process in which the information is exchanged via electronic media in the sense of Schmid (1998).

This broad definition covers a wide variety of mechanisms. Automated negotiations are a subfield of electronic negotiations which cover the entire negotiation process, including the final decision about an agreement or a disagreement. These decisions are made without human interaction – purely software agent based.

In the following paragraphs different negotiation approaches which have been proposed by several authors are summarized.

Chaudhury et al. (1991) gave an early approach to competitive negotiation scenarios. They focussed on the gaining interest in the development of computer systems for use in negotiations. They state that current systems are exclusively focussed on cooperative problem solving. Their work concentrates on negotiation situations that are marked by conflict. Using a social-theoretic framework they identify the scope of computer assistance for negotiation in non-cooperative situations. With this framework they contribute an early negotiation support system.

The authors Benyoucef and Keller (2000) focus on a clear description of the rules that govern the existing diversity of negotiation types. They state that "negotiation rules" denote the rules governing the negotiation, whereas "negotiation strategy" describes the rules used by an individual participant during the negotiation in order to make the best out of her participation. The research described in their approach focuses on combined negotiations, and concepts and architectures to support them. They state that there are five desiderata on automated negotiations:

(i) Correct: an automated negotiation must not contain errors such as deadlocks or incorrect handling of exceptions.
(ii) Reasonable: it must allow adequate time for bids to be made.
(iii) Robust: it should continue to function properly after an improper action by the user.
(iv) Fast: it has to execute and respond quickly.
(v) Tracetable: An automated negotiation must also be tractable.

With their negotiation formalism they refer to the Object Management Group (OMG)[3] negotiation facility. They are capturing the negotiation process using finite state machines (FSM) combined together with state diagrams adopted by UML.

In Benyoucef et al. (2001) they are focusing on combined negotiations as a novel and general type of negotiation, in which the user is interested in many goods or services and consequently engages in many negotiations at the same time. The authors state that negotiations are independent of each other, whereas the goods or services are typically interdependent. They define a combined negotiation support system (CNSS) assisting users to engage in combined negotiations. They designed such a CNSS called CONSENSUS which relies on workflow technology, negotiating software agents, and rule engine technology. The originality of that architecture is the fact that the user models combined negotiations at build time using workflows that capture the sequencing of the individual negotiations and the dependencies between them.

Furthermore in Benyoucef et al. (2001) the authors focus on infrastructures for rule driven negotiating software agents. They are aiming to define mechanisms for adaptable negotiation strategies. The fact that in most automated negotiation infrastructures the strategies of the agents are hard coded, motivates their view. They define the INfraStructure for rULe-driven negotiating

[3] http://www.omg.org (06/25/2002)

software Agents (INSULA) which is based on the representation of declarative knowledge implemented in if-then rules. These can be exploited by software agents using an inference engine.

Benyoucef et al. (2000) introduce the issue of a Generic Experimentation Engine (GEE) which is designed to investigate the various research questions concerning eNegotiations. It is part of the Towards Electronic Marketplaces (TEM) project, a joint industry-university project. The GEE supports game oriented experimentation and allows the study of human behavior under various game restrictions.

An interesting discussion is introduced by Kersten et al. (2000) who ask: "Are all e-negotiations auctions?" They state that auctions, which can be viewed as distributive negotiation protocols, appear to be irrelevant in cooperative contexts. In business-to-business commerce, the participants are often less concerned with price and more with relationships. Negotiation-like protocols are dominating these circumstances. This fact is especially true when focussing on inter-business relationships. For that reason the question is answered negatively: Auctions are a subset of all electronic negotiations, especially a subset of the competitive negotiations.

Guttman and Maes (1999) distinguish distributed and integrative negotiations using autonomous software agents for automation. They focus on integrative negotiations for the retail electronic commerce. The impact of multi-attribute utility theory, distributed constraint satisfaction and conjoint analysis to support automated integrative negotiations. They conclude that from a customer's perspective, integrated negotiation help to compare merchant offerings across their full range of value resulting in mutually rewarding hassle free shopping experiences. This distinction is drawn towards the comparison of cooperative versus competitive negotiations in Guttman and Maes (1998). One of their key results is the evaluation of Distributed Constraint Satisfaction Problem (DCSP) protocols to best support todays (and likely tomorrows) retail market model.

In Guttman et al. (1998a) and Guttman et al. (1998b) the authors focus on comparing several agent based electronic commerce systems describing their Consumer Buying Behavior (CBB) model. They discuss a variety of Artificial Intelligence (AI) techniques that support agent mediation and conclude with future directions of agent-mediated electronic commerce research. They summarize that in next generation electronic commerce systems agents will enhance customer satisfaction and streamline business-to-business transactions mainly by reducing transaction costs.

Ueyama et al. (2001) introduce an electronic commerce platform using a bilateral electronic negotiation model for bargaining between two participants. They use Grasshopper mobile agents for the representation of the negotiation parties. They also follow the OMG[4] bilateral negotiation protocol. They focus on similarity computation among offers and requests using XML representations and feature-space similarity measures.

Zlotkin and Rosenschein (1996) develop an Unified Negotiation Protocol (UNP). They analyze negotiation situations where agents have incomplete information on the goals of other agents and show what goal declaration strategies the agents might adopt to increase their utility. They introduce two mechanisms, one strict, the other tolerant, and analyze their aspects on the stability and efficiency of negotiation outcomes.

In Morris and Maes (2000) the authors show the limitations of current electronic auction systems. They present guidelines which allow a more dynamic attribute-based bidding which allows to specify preferences beyond the mere bid price. They develop a prototype which allows multi-attribute auctions in the airline fright domain.

The authors Teich et al. (1999) and Teich et al. (1999) focus on simple multiple-issue algorithms and heuristics that could be used in electronic auctions and electronic markets, to match business-to-business and consumer based dovetailing underlaying interests and preferences. They argue that such dovetailed matches should help to stabilize markets and make them more efficient. Their key motivation is the reduction of the likelihood of damaging price wars in future global markets.

In Hoffner et al. (1999) the authors address an investigation into the distribution issues surrounding the design and implementation of virtual marketplaces. With a distributed approach they also focus on the problem of market provider legacy integration. They design and implement a virtual insurance market using Java and CORBA technology. The key issue of this contribution is the technical aspect of distributed electronic markets and marketplaces.

The authors Koppius (1998) and Koppius et al. (2000) address the term of multidimensional auctions. They focus on combinatorial auctions which they assign the term multidimensional. This view differs from the definition of "multidimensional" which is presented by Bichler et al. (2002). Bichler and coauthors define the term "multidimensional" as an umbrella term, covering multi-unit, multi-item and multi-issue/multi-attribute auctions. This view again differs from the definition of the term "multidimensional" given in this work. Here, as already defined in Section 2.1.2 "multidimensional" defines an enhancement to the definition of "multi-attribute", where attributes may be dimensions with subdimensions and substructures.

Lomuscio et al. (2001) and Jennings et al. (2001) provide a scheme for classifying negotiation in electronic commerce. They see many current business-to-customer and business-to-business electronic commerce applications to be

[4] http://www.omg.org (06/25/2002)

nothing more then electronic catalogs on which users can choose products for a fixed price. Doing this, they state that the potential of electronic commerce are by far not exploited. They focus on the prediction of the emerging of systems in the next years which are based on automatic negotiation. They identify several parameters on which automatic negotiation is depending on:

1. The cardinality of the negotiation: single-issue or multiple-issue, one-to-one or one-to-many or many-to-many.
2. The agent characteristics: Role, rationality, knowledge, commitment, social behavior and bidding strategy.
3. Environments and goods characteristics: Private/public value of the goods, nature of the goods.
4. Events parameters: Bid validity, bid visibility, clearing schedule and time-outs, quotes schedule.
5. Information parameters: Price quotes, transaction history.
6. Allocation parameters: They only apply in many-to-one and many-to-many scenarios. They identify the winner in case there is more than one agent interested in the good.

The authors see their classification framework as a step towards more sophisticated electronic negotiations platforms, featuring agent-based automatic negotiations in future years. The interdisciplinary research integrating economics, computer science as well as Law is currently strongly gaining importance. In Section 3.2.3 further approaches from computer science are introduced. However, because many researchers in computer science focus on negotiation approaches, too, intersections to the economic approaches will be obvious.

3.2.3 Electronic Negotiation Approaches with Multi-Agent Systems Focus

In this section, the paradigm of Multi-Agent Systems (MAS) is focussed. The MAS definitions which have been given in Section 2.2.3 above are underlain with current research done in this field. A good overview on multi-agent research within the last years is given by the authors Wooldridge and Jennings (1995), Wooldridge et al. (1996) and Müller (1997). Besides that the authors Jennings et al. (1998) provide a roadmap of agent research. Finally Weiss (1999) characterizes the multi-agent research and provides the MAS domain as a new perspective on distributed artificial intelligence (DAI) research. Due to the fact that the basic agent research questions are not the key issue of this work, the given sources are provided to support definitions and current questions in detail.

Many approaches in electronic negotiation and matchmaking research which have been introduced in this chapter do already focus on the paradigm of autonomous agents. Approaches in electronic negotiation research – as shown in Section 3.2.2 show significant effort to automate negotiations using results from this discipline. The Multi-Agent paradigm is ideally suited to be applied in electronic negotiation scenarios. Within the last years the disciplines of electronic market research and Multi-Agent Systems research have met. Researchers from DAI and MAS fields applied their results to electronic markets and electronic negotiations and researchers from economics applied MAS to implement market mechanisms.

Lux and Steiner (1995) introduced one of the first multi agent platforms called Multi-Agent Environment for Constructing Cooperative Applications (MECCA). This environment, which is based on a standard library for the design of multi-agent systems uses KQML (see Finin et al. 1994) as communication language. Together with other researchers in this research area, the authors of this work started the definition of a common agent communication language (ACL). This agent communication standard which led to the Foundation for Intelligent Physical Agents (FIPA)[5] standard. The FIPA standard is designed by a group of researches and practioners who first met in 1996 and released the first ACL standard in 1997. The latest release is the FIPA 2000 standard which is used by many agent architectures as communication language.

Bellifemine et al. (2001) defined the Java Agent DEvelopment Framework (JADE) which is a generic agent development framework based on JAVA. The authors Poslad et al. (2000) defined and implemented the Foundation for Intelligent Physical Agents – Operating System (FIPA-OS). This system provides another approach to realize a generic Multi-Agent System which can be integrated into other multi-agent systems as well as used for an implementation of a domain application. Both approaches – the JADE and the FIPA-OS approaches are driven by the research direction of defining a generic agent based library for the implementation of prototypes and applications.

Besides the definition and implemetation of standardized agent development frameworks, the modelling of large Multi-Agent Systems is advancing by the integration of software agent contributions into UML modelling language approaches. Within the domain of developing such software agent architectures UML based modelling approaches are e.g. *Agent UML* which enable a standardized construction of agents. The authors Bauer et al. (2001) focus this issue and propose a new field.

Varian (1995) uses "computerized agents" as actors for the design of economic mechanisms. He focuses on auctions as a mechanism to determine the "maximum willingness to pay" of an individual. The notion of "hyper-rationality" which the author attaches to game theoretic decision making is ideally suited to describe the decision capability of software agents. These agents have a much higher computational power then human beings and therefore decide

[5] http://www.fipa.org (01/01/2002)

more rational. He concludes that computerized agents and so software agents are ideally suited to map economic mechanisms into automated applications.

In Oliver (1996b) and Oliver (1996c) as well as Oliver (1997) the author describes a framework based on evolutionary computation methods for artificial adaptive agents. The key idea is to prove that autonomous agents do not leave as much money on the table by bidding pareto-inferior compared to human bidders. The application domain which the author investigates are multi-issue auctions.

Also, Calisti and Faltings (2001) consider the issue of agent based negotiations. Here, multi-provider interactions are focussed underlaying theories from agent communication and coordination.

In Guttman et al. (1997) the authors focus on a framework which enables agent-mediated integrative negotiations. They apply techniques from multi-attribute utility theory, distributed constraint satisfaction, and conjoint analysis to improve the negotiation behavior of computational agents.

Sierra et al. (1999) see automated negotiation as a joint decision which is made by two or more parties. They define an agent negotiation architecture which enables individual reasoning mechanisms for agents. Due to the general inability of computerized agents to make crisp decisions they propose fuzzy similarity measures to extend the so called "classical negotiation model".

In Sandholm (1999) and Sandholm (2000) the author introduces the eMediator approach. The eMediator – a next generation electronic commerce server – enables algorithmic support and game theoretic incentive engineering for electronic negotiations. The author focuses on combinatorial auctions in which bidders can place bids on combinations of several items. This allows to express complementarities between items instead of speculating about the impact of getting other complementary items.

In Faratin et al. (1999), Faratin et al. (2000) as well as Faratin (2000) the authors introduce mechanisms for automated service negotiation among autonomous computational agents. The approach to service negotiation is not limited to services or goods in the economic sense but moreover approaches services among software agent entities in general. They approache the topic from a game theoretic as well as computational negotiation model view. In the work a service oriented negotiation model is introduced which regards the following dimensions.

- Interaction protocols
- A bilateral negotiation model
- Responsive and deliberative mechanisms
- The responsive mechanism
- The tradeoff-mechanism

- The issue set manipulation mechanism
- The meta strategy mechanism

Using these mechanisms a negotiation framework is designed and empirically evaluated in strategic and non-strategic experiments.

3.2.4 Electronic Auctions

In this section a short excursion towards electronic auctions is given. The major protocols are introduced. Some authors which have been presenting key-research in this domain are provided.

Weinhardt (2002), Kumar and Feldman (1998), Koppius (1998) and Koppius et al. (2000) provide a definition of the notion of electronic auctions also referred to as Internet auctions. Weinhardt (2002) sees Internet auctions as an economic mechanism which provides a digital version of allocation mechanisms. Electronic auctions define – analog to regular auctions – an economic mechanism which is designed to solve competitive resource allocation problems. Generally said, an electronic auction is the implementation of auction protocols using electronic media.

Using electronic media to apply auction protocols several advantages arise. In electronic markets, many dimensions can be considered which are too complex to express in non electronic negotiations. Bichler et al. (1999) state that the item characteristics are an important factor for determining the type of appropriate negotiation and matchmaking mechanism to be applied. There are several terms which provide a definition framework for the design of negotiation characteristics referring to concrete item characteristics.

1. Multi-phased Auctions: Several phases are carried out determining the auctions outcome.
2. Multi-stage Auctions: Similar to multi-phased auctions, several stages have to be passed before the auction terminates. In this case the order of the stages is relevant.
3. Multi-unit Auctions: Multi-unit auctions describe auctions in which several units of the same object are auctioneered.
4. Multi-item Auctions: Multi-item auctions describe auctions in which several – possibly heterogeneous – items are auctioneered.
5. Multi-attribute Auctions: As Bichler (2000b) and Bichler (2000a) define multi-attribute auctions as auctions on which the over all score computation is not limited to bids on the mere price. Several aspects of a good can be bid on. Commonly a virtual currency is introduced to provide the over-all score which is then, again, mapped to a price.
6. Multidimensional Auctions: Bichler and Werthner (2000) and Bichler et al. (2002) see multidimensional auctions as an umbrella term for multi-unit, multi-item and multi-attribute auctions.

Matchmaking plays a crucial role within electronic auctions. Within each bidding procedure a winner has to be determined. In single-attribute auctions where only the price can be bid on, the highest price wins. In this case no sophisticated matchmaking mechanism has to be introduced. In multi-attribute auctions, where several attributes are provided which can be bid on as well as mechanisms are needed to compute an overall score, matchmaking mechanisms are required. Generally said, the more attributes and sub-attributes are provided – i.e. the more complex the bid structure is defined – the more complex matchmaking procedures have to be introduced. Veit et al. (2002b) refer to the matchmaking problem in electronic negotiation as a relevance computation mechanisms between structured objects, describing offers and requests.

3.3 Discussion

In this chapter many approaches concerning matchmaking and electronic negotiations are introduced. Within the matchmaking models, several trials to define generic frameworks were undertaken. Table 3.5 shows a comparison of some electronic negotiation and their matchmaking mechanisms, as far as they were described in literature. Table 3.6 shows a comparison among the introduced pure matchmaking architectures.

Within the SILKROAD (see Ströbel 2001b) framework the matchmaking component is based on constraint satisfaction (similar to the EMC matchmaker by Ströbel and Stolze (2001), see below). The offers and requests can be supported in a subject based structure. This enables a wide variety of application domains. On the other hand the matchmaking logic is limited.

The OFFER framework (see Bichler et al. 1998b) provides matchmaking via the "Lookup" interface. This matchmaking allows the discovery of objects based on the services they provide. The matchmaking is based on constraints which are compared using certain rules of certain Object Trader implementations such as JTrader (a GNU-licensed JAVA implementation of the Object Trader Service, the Trading Object Service is a CORBA standard since mid of 1996). Again, the constraint based matchmaking limits the application logic to a mediocre level.

The INSPIRE system (see Lo and Kersten 1999) provides no concrete matching unit. Moreover, it provides communication support among offering parties and requesting parties to submit the individual preferences. The matchmaking is performed by the parties, accepting or rejecting an offer or request. The advantage of this system is the openness in negotiating the position. But, because concrete functions for machine based matchmaking are missing, the aim of the system is not to take over the matchmaking completely but to support a matchmaking assistance to the participating parties.

INSULA (see Benyoucef et al. 2001) provides a rule based matchmaking unit. Several attributes can be supported in a domain specific way. These attributes are then matched using the constraints of the attributes of the counterpart applying the matching rules. This design limits the matchmaking complexity but enables domain specific attributing. Anyway, the matchmaking task is not a core aspect of the INSULA approach.

Now, systems which provide purely the matchmaking procedure and do not support negotiation elements are compared (see also Table 3.6).

The EMC framework by Ströbel and Stolze (2001) contains a matchmaking unit which allows free definable offer and request structures. These structures are application dependent and adaptable. A disadvantage which limits the application domains for this framework is the fact that the offer and request attributes are matched only based upon constraints and no discrete values are allowed.

Approach	**Offer/Request genericity**	**Agreement rule**
SILKROAD, Ströbel (2001b)	subject based	adjusted winner procedure
OFFER, Bichler et al. (1998b)	no structure, single attribute	different auction protocols
INSPIRE, Lo and Kersten (1999)	message based, unstructured	bilateral decision support functions
INSULA, Benyoucef et al. (2001)	subject based	explicit rules defined

Table 3.5. Comparison of Negotiation Approaches

The SHADE approach by Kuokka and Harada (1996) defines one of the first generic free text matchmakers. This system has the advantage to provide distance functions from information retrieval which make it fairly flexible and domain independent. The main disadvantage of this system is the fact that it does not provide mechanisms to match structured offer and request. This limits its application in practice.

The COINS matchmaker by Kuokka and Harada (1996) is defined to enable the automation of procurement procedures in plants. This matchmaker provides several functions for the application in heterogeneous domains but incorporates only static offer and request structures. These static structures are defined for one application domain and are not designed to be adaptable to other domains.

The SCA matchmaking architecture by Weinstein and Birmingham (1997) is based on a specific auction protocol. There exists one central Service Classi-

Approach	**Offer/Request genericity**	**Domain independent applicability**
EMC, Ströbel and Stolze (2001)	structure free definable	only constraint based matchmaking
SHADE, Kuokka and Harada (1996)	no structure	only free text (IR methods)
COINS, Kuokka and Harada (1996)	fixed structure	engineering domain
SCA, Weinstein and Birmingham (1997)	manual descriptions	domain independent but only exact matches
LARKS, Sycara et al. (1998)	fixed, simple structure	domain independent, several sequential matchmaking steps
IMPACT, Subrahmanian et al. (2000)	very simple structure (two verbs one noun)	domain independent verb and noun hierarchies
InfoSleuth, Nodine et al. (1999)	no structure	rule based reasoning
Grappa, introduced in this work	*highly defineable structure*	*domain independent, any distance function/metric integrable*

Table 3.6. Comparison of Matchmaking Approaches

fication Agent (SCA) which classifies requests and offers. The classification algorithm works on unstructured descriptions and hence, does not enable structured offers and requests.

LARKS by Sycara et al. (1998) is one of the most powerful matchmaking approaches known so far. It provides several sequential matchmaking processes, focussing different aspects. Still, the application domains are limited due to the fact that the offer and request structure is static. The sequential matchmaking processes enable a high matchmaking quality. For a new matchmaking approach it is interesting to integrate this sequential process as several parallel processes.

The IMPACT matchmaker by Subrahmanian et al. (2000) is based on a very simple offer and request structure. It allows only VerbNounterms, consisting of two verbs and a noun as introduced above as offer and request structures. On the other hand, the term hierarchies enable a powerful matchmaking which might also be applied in specific domains. Still, it is limited to simple applications due to its fixed offer and request structures.

In the InfoSleuth approach Nodine et al. (1999) the syntax and semantics of offers and requests is matched. There exists no structure for offers and requests to encapsulate those. The reasoning is included in a rule basis which enables the broker and matchmaker to determine relevances. Again the domain independent application is limited by the fact that no offer and request structures can be defined.

The limitations especially in dynamically integrating heterogeneous offer and request structures of the approaches discussed so far give motivation for the definition of a generic matchmaking framework. This definition is undertaken with GRAPPA which tries to combine the benefits of a generic approach with the key advantages of domain specific solutions. The GRAPPA framework is explicitly defined to enable flexibly generated offer and request structures as well as to be completely domain independent by supporting distance function and metric interfaces which allow the easy integration of domain dependent and generic functions.

3.4 Summary

In Chapter 2 the research areas as well as term definitions from different researchers views within economics and computer science are shown from an overview perspective. In this chapter the related work relevant to the GRAPPA approach is examined in detail. In Section 3.1 theoretical and practical approaches to multidimensional matchmaking problems mostly in multi-agent systems are introduced. This section provides the most detailed formal definitions throughout the related work chapter. Special attention is payed to the approach by Sycara and coworkers due to the fact that the LARKS language with the matchmaking approach in the RETSINA project has given major inputs to the GRAPPA framework.

A comprehensive presentation of many concrete electronic negotiation and matchmaking projects from research and practice is provided. The chapter opens with Section 3.2 in which concrete applications from electronic negotiation are introduced. Here, special attention is spent on multidimensional negotiations and the Montreal Taxonomy. A short excursion to auction theory provides the basic concepts from auctions as market mechanisms.

Finally in Section 3.3 the different frameworks, projects and approaches are compared using several characteristics. This comparison works out the limitations of the provided systems and motivates the definition of the GRAPPA framework.

This chapter provided a description of many research projects in the field. The GRAPPA approach which is formally defined and introduced in the next chapter is influenced by many of these projects. It still exceeds the genericity of the known projects and provides a very open and flexible definition structure for matchmaking applications in market designs.

4 Matchmaking Architecture

In this chapter the mathematical foundation for the GRAPPA matchmaking framework is layed. The aim of the chapter is to introduce a well defined, sound and consistent model which is a suitable basis for a computational matchmaking framework.

Section 4.1 introduces the concepts underlying the matchmaking model. This section is designed to provide an over-all understanding of the model and the structure of the consecutive sections. Section 4.2 provides an introduction to the components of the GRAPPA matchmaking framework. Section 4.3 to Section 4.5 then introduce the concrete definitions as well as metrics and distance functions for matchmaking implementation which are the columns of the GRAPPA matchmaking framework.

4.1 Concepts

For the modelling of multidimensional matchmaking a role model is needed. Within the GRAPPA framework three roles are defined.

- The candidate
- The centroid
- The matchmaker

These semantic roles are assigned to actors – in the case of this framework agents – in Section 4.4. The definitions of the term candidate, centroid and matchmaker were given in Chapter 2. The central idea in multidimensional matchmaking is the modelling of candidates and centroids as information objects which consist of more then one dimension. This makes it possible to semantically differentiate between several aspects of an object which is ranked/negotiated. The matchmaker itself is then able to apply different functions to the dimensions in order to compute the relevance.

The central aspect in computing a relevance between an candidate and a centroid is the definition of distance functions. Distance functions and metrics have the property to take as input two instances of a domain (e.g. real numbers, free texts, images, etc.) and compute a value between 0 and 1 as a distance value reflecting the distance between the two instances.

The mathematical definition of a distance function or metric allows that the result is even a positive real value (i.e., a value $\in \mathbb{R}_0^+$). As shown in this chapter there is a mapping between the interval [0; 1] and $\mathbb{R}_0^+$ which is order preserving so that consequently it is identical to consider the interval [0; 1]. The semantic of a distance function or metric implies that a value close to 1 reflects a big distance, i.e., the instances do not match and a value close to 0 reflects a little distance, i.e., the instances match well. A mapping is defined which inverts the value domain of the distance function and generates a function reflecting the relevance of one instance towards another. This – so called relevance function – has also the value domain [0; 1] but here, 0 means bad matching and 1 means good matching. This valuation reflects the human judgement better – i.e., a good score is close to 100 percent, a bad one close to 0 percent.

Due to the fact that it is mathematically favorable to use distance functions and metrics, the mapping from distance functions towards relevance functions is done in the last step within the implementation and is not considered in the next sections.

In Section 4.3 special aspects in modelling multidimensionality in real-world matchmaking problems are discussed. In many application scenarios candidates and centroids do not have the same structure. This is regarded within the Grappa framework, since, Grappa allows heterogeneous candidate and centroid structures. These can be mapped on each other using a mapping function. To keep the modelling clear and understandable for the reader, within Section 4.4 the Grappa matchmaker is firstly modelled using homogenous candidate and centroid profiles in Section 4.4.1. In the following Section 4.4.3 the mapping between candidate and centroid structures in Grappa is defined.

It is also central to apply distance functions and metrics which reflect the realistic matching judgement of human "matchers" well. Therefore different functions are applied. The functions from information retrieval (IR) play a central role which are in depth introduced in Section 4.5.2.

In multidimensional matchmaking scenarios it is of course not enough to simply compute the relevance of the attribute instances among each other. Moreover it is crucial to define appropriate complex distance functions, also called aggregate functions to compute dimension- and overall-scores of one candidate instance towards a centroid instance. These complex distance functions and their weighting paradigms are introduced in Section 4.5.4 and 4.5.5.

4.2 Generic Matchmaking Framework – Overview

The aim of this section is to outline the structure of the multidimensional matchmaking framework Grappa. This outline references the later chapters

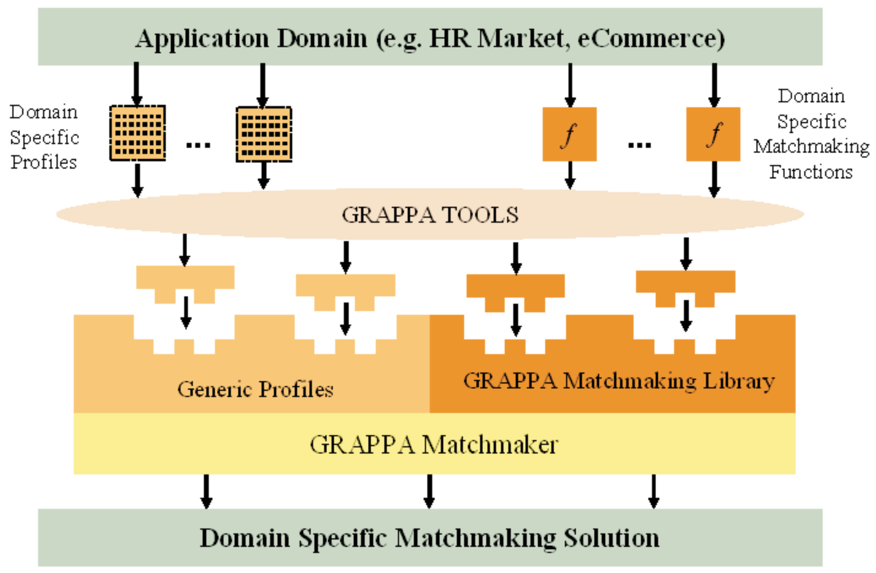

Application Domain (e.g. HR Market, eCommerce)
Domain Specific Profiles
Domain Specific Matchmaking Functions
GRAPPA TOOLS
Generic Profiles
GRAPPA Matchmaking Library
GRAPPA Matchmaker
Domain Specific Matchmaking Solution

Figure 4.1. GRAPPA Matchmaking Framework

and enables an overview on the detailed definitions and implementations considered in this work. Figure 4.1 illustrates the structure of the GRAPPA matchmaking framework. The concepts and definitions of the key terms used in this structure are presented in the following sections.

The GRAPPA matchmaking framework consists of three major parts. Its core is the matchmaking engine described in Section 4.2.1. It is complemented by the matchmaking library in Section 4.2.2 and the matchmaking toolkit in Section 4.2.3.

4.2.1 GRAPPA Matchmaking Engine

The matchmaking engine accepts a set of candidates and one centroid as input. The candidates which have to be provided as instances of the matchmakers offer information object structure are either provided by a candidate which is directly attached to the matchmaker or – in case the matchmaker is only adressed as a service – retrieved from different data sources. The centroid which has to be provided as an instance of the matchmakers request information object structure is matched against each of the candidate information object instances.

The offer as well as the request information object structure are possibly multidimensional. They may consist of complex types constructed from a domain specific set of atomic types under application of four complex type constructors: list, array, record and set. The overall distance, a real value between 0 and 1, is obtained by recursively computing the distance values for different profile subdimensions, as shown in the example in Chapter 6, and propagating them upwards to compute the values for their parents. In Section 4.2.2 these concepts are defined in detail.

For the attributes, the specific distance function for the particular type is applied and the result is propagated upwards. Then, at the next higher level, all atomic distances between the attributes in this level are merged to one distance value for this complex type under application of *complex distance functions.*

The result of the recursive computation of distance values is

(i) an overall distance (real value between 0 and 1) which reflects the quality of the considered candidate for the current centroid instance.
(ii) a structure which consists of the individual distance results in each layer.

These results implement the desiderata of the matchmaking definition given in Section 2.1.2. The best k candidates with respect to the current centroid are returned as the result of the match. This list is ranked using the value from (i). The agent, or the agent's owner, can then recur into the structure described in (ii) to obtain an explanation how the particular overall result arose, e.g. , which aspects of the match contributed to a good or bad overall result.

4.2.2 Grappa Matchmaking Library

The Grappa matchmaking library hosts an extensible collection of predefined candidate and centroid structure and general purpose as well as domain specific distance functions. The profile schemes can be used as a basis for application specific profiles; the distance functions provide uniform interfaces that allows to flexibly combine them to develop specific matchmaking solutions.

It is essential for a matchmaking system to provide powerful distance functions. Currently, distance functions for FreeText, Keyword, Interval, TimeInterval, DateInterval, Boolean, ProperName and Number atomic values (i.e. instances of atomic types) are provided. All of these distance functions have the property to take two atomic values as input and to provide a real number between 0 and 1 as output (distance). Additionally domain specific distance functions, can be integrated. On top of these atomic functions, *complex distance functions* are defined. Currently, WeightedAverage, Average, Minimum, Maximum are supported as predefined *complex distance functions.* As for atomic distance functions, it is possible to define domain specific *complex distance functions* and integrate them into a domain specific matchmaker.

4.2.3 Grappa Toolkit

The Grappa Toolkit provides a set of tools which enable the development of a multidimensional matchmaker for specific applications mainly through configuration without much coding work. To guide the marketplace or negotiation designer a 5-step process to obtain a domain specific matchmaking solution is defined as follows:

1. Define the centroid and candidate structures (atomic entries).
2. Define the clusters of attributes (pseudo-orthogonalization). Clustering can be recursive. A set of clusters is adding up to a dimension.
3. Associate the dimensions of the centroid with the dimensions of the candidate by applying appropriate distance functions.
4. Combine the results of the distance functions to an overall distance value (e.g. , weighted sum).
5. Apply feedback regarding the quality of the matches, e.g. , by adaptively changing weights or matching functions.

The feedback process in (5) is implemented as a simple real value between 0 and 1 which the systems user can return to the system. With this value the user can indicate how he judged the matchmaking result for the particular item.

In the following the general structure and the granularity of the Grappa matchmaking approach are introduced. In the following the formal definition of the framework is described.

4.3 Modelling Multidimensionality

If candidates and centroids are non-structured collections of content, no modelling questions arise. One of the key aspects of this work is to improve the matchmaking quality by structuring objects using dimensions and attributes. A candidate or centroid object is no longer a flat document but a structured object.

To provide a structure for an information object different aspects have to be regarded. The dimensions, which are themselves collections of subdimensions and attributes have to be selected. Each dimension is supposed to represent a specific semantic aspect of the candidates which will be described by the instances of the structure. Selecting these aspects which are bundled in dimensions it is essential to provide preferential independence between the dimensions. The following definition introduces the notion of preferential independence and preferential dependence as it is used in this work.

Definition 4.3.1 ***Preferential Independence and Dependence***

Let δ_1, δ_2 be matchmaking attributes of a matchmaking object O. Then, δ_1 and δ_2 are called **preferentially independent**, written

$$\delta_1 \not\approx \delta_2$$

iff any valid parameter value $s_1 \in D(\delta_1)$, where D is the domain of δ_1, does not influence or entail the parameter value $t_1 \in D(\delta_2)$.
Does $\delta_1 \not\approx \delta_2$ not hold, δ_1 and δ_2 are called **preferentially dependent** ($\delta_1 \approx \delta_2$).

If two attributes are preferential independent that means that for all possible instances, in case the content of one attribute changes, the content of the other attribute stays unchanged. The higher the number of dimensions and attributes for a concrete application is, the higher the probability rises, that certain dimensions become preferential dependent. For that reason it is recommendable to introduce a weaker notion then the preferential independence.

For application design using the Grappa framework an orthogonalization phase is applied during the "clustering of attributes" phase as shown in Chatper 2. This orthogonalization is used to provide an object structure which fulfills preferential independence. In practice it can not be avoided to have semantic overlappings between the attributes in all cases. For this reason a so called pseudo-orthogonalization is performed. This process simply consists of a phase in which the attributes are multiplied and forced into an orthogonal structure which may not be orthogonal in the strict sense.

4.4 The Grappa Framework

In this section the Grappa is introduced. Grappa is defined in a way to allow multidimensional matchmaking over various kinds of data.

Therefore three different role behaviors for agents are defined:

- The *candiate:* The candidate is a passive provider agent (PPA) which provides its goods or services to other agents.
- The *centroid:* The centroid is a requester agent (RA) which can only fulfill its task by collaborating with a candidate.
- The *Multidimensional Matchmaker Agent (MMA):* This agent stores candidates of PPAs and answers with a list of candidates to centroids from a centroid.

The aim of Grappa is not to provide a complete multi-agent system but rather to give a generic matchmaking approach that can be plugged into any existing agent system by defining an agent MMA as well as supporting this agent with a structured GRAPPA matchmaking base. Then a candidate is any agent that provides a full description of its candidates denoted in the Offer and Request Description Language (GRAPPA ORDL) to the MMA. This candidate must be given using the Offer ODS of the structured GRAPPA matchmaking base which the MMA is supported by. It is possible to construct a wrapper agent that wraps any candidates of any offer-request language (like KQML) into GRAPPA specific candidates, which is specified using a subset of XML.

Firstly, a formal language in which candidates and centroids can be defined must be specified.

4.4.1 The Offer and Request Description Language (ORDL) of GRAPPA

In this section the ORDL used by GRAPPA is defined. Before any specific language is defined some desiderata for such a language are formulated.

Desiderata for an ORDL An ORDL for a multidimensional matchmaker should have the following capabilities:

- **Easy to use:** The ORDL should be easy to use. A user who desires to put his agent into the system should be able to identify the inputs without additional explanation. The language should be readable and writeable by any user.
- **Easy to process:** The specified language should be easy to process, present and store.
- **Easy to adapt:** The language should be easy to adapt to applications. There must be a basic set of descriptors which are always used.
- **Easy to expand:** The language chosen should to be expandable. It should be easy to include new types as well as new distance functions.

GRAPPA **Matchmaking System** To proceed in defining an ORDL, types of which candidates and centroids consist must be specified.

Postulate 4.4.2 ***Types in*** GRAPPA

1. There is a set of basic types
$$\mathcal{B} := \{NAME_1, \ldots, NAME_m\}.$$
2. There exists a set of types which includes the basic types
$$\mathcal{T} \supset \mathcal{B}$$

Basic types are used to express information encapsulated in attributes. Each attribute used in an candidate or centroid is of a certain basic type. Based on the postulated set of types $\mathcal{T}$, it is possible to declare type definitions.

Definition 4.4.3 ***Set, List, Array, Record***

1. A **set** of elements is an unordered collection of elements of the same type.
2. A **list** of elements is a consecutive row of elements of the same type which ends with a terminating symbol (NIL). This row has arbitrary length.
3. An **array** of elements is a consecutive row of a fixed number of n elements of the same type.
4. A **record** of elements is a consecutive row of a fixed number of n elements of possibly different types.

The terms which have been introduced so far enable the notion of a type definition.

Definition 4.4.4 ***Type Definition***

A **type definition** (TD) of a new type $NAME$ is an expression of one of the following four forms. $NAME$ is used as a wildcard for the new type to be defined. The constructors are to be used exclusively alternatively.

(i) $NAME := \{NAME_i\}$ where $NAME_i \in \mathcal{T}$. $\{NAME_i\}$ denotes a **set** of arbitrary size, whose elements are of type $NAME_i$.
(ii) $NAME := NAME_i^+$ where $NAME_i \in \mathcal{T}$. $NAME_i^+$ denotes a **list** of arbitrary length, whose elements are of type $NAME_i$.
(iii) $NAME := NAME_i[1 : n]$ where $NAME_i \in \mathcal{T}$. $NAME_i[1 : n]$ denotes an **array** of length n, whose elements are of type $NAME_i$.
(iv) $NAME := (NAME_1, \ldots, NAME_k)$ where $NAME_i \in \mathcal{T}$ for $i = 1, ..., k$.
$(NAME_1, \ldots, NAME_k)$ denotes a **record** of length k where $NAME_i \in \mathcal{T}$ for $i = 1, \ldots, k$.

In order to provide a sound framework certain criteria have to be fulfilled to make a type definition valid. The level function l is a recursively defined function, computing the type depth. The type depth is the number of sub-dimensions which are encapsulated before reaching the attribute level and is very important for the implementation of the framework.

Definition 4.4.5 ***Valid Type Definition***

A valid type definition for $\mathcal{T}$ is a set TD of type definitions with the following properties:

(i) For all $\tau \in \mathcal{B}$ there is no definition in TD.
(ii) For all $\tau \in \mathcal{T} \backslash \mathcal{B}$ there exists one and only one definition in TD.
(iii) There exists a mapping $l : \mathcal{T} \to \mathbb{N}$ such that

1. For all $\tau \in \mathcal{B}$ $l(\tau) = 0$.
2. Let τ result by applying a complex type constructor to types $\tau_1, \ldots, \tau_n \in \mathcal{T}$, where $n \geq 1$ (i.e. $\tau \in \mathcal{T} \backslash \mathcal{B}$. Then l is recursively defined as

$$l(\tau) = 1 + \max_i l(\tau_i)$$

Now basic types and valid type definitions are known. To enable the definition of a rich, multidimensional matchmaking object structure, the notion of a complex type is defined as follows.

Definition 4.4.6 ***Complex Type***

A type $\tau \in \mathcal{T} \backslash \mathcal{B}$ is called **complex type**.

For every complex type $\tau \in \mathcal{T} \backslash \mathcal{B}$ the level function $l(\tau)$ is greater than zero.

Definition 4.4.7 ***Valid Type Set***

A valid type set is a pair of the form

$$(\mathcal{T}, TD)$$

where
1. $\mathcal{T}$ denotes a set of types.
2. TD denotes a set of valid type definitions for the complex types in $\mathcal{T} \backslash \mathcal{B}$.

Using the types defined in the definition above a candidate and centroid description scheme can be defined. The next step is to define how this generic candidate or centroid description scheme looks like. Candidate and centroid description schemes formalize the information object structure of candidates and centroids formally.

Definition 4.4.8 ***Offer* or *Request Definition Scheme (ODS, RDS)***

An **offer** or **request definition scheme** is a record of valid types which characterize a class of candidates or centroids sufficiently:

$$(\psi_1, \ldots, \psi_n)$$

where $\psi_i \in \mathcal{T}$ for $i = 1, \ldots, n$.

As mentioned above, candidates and centroids may be of heterogeneous structure. In Section 4.4.3 a general mapping among heterogeneous candidate and centroid structures is introduced.

On the way to the definition of a structured Grappa matchmaking system, the definitions of atomic (basic) and complex values are necessary.

Definition 4.4.9 ***Atomic Value, Complex Value***

Let $\tau \in \mathcal{T}$ be any type. Let $D(\tau)$ be the set of all instances of the type τ called **domain of type** τ.

1. If τ is a basic type, then every $v_i \in D(\tau)$ is called **atomic value**.
2. If τ is a complex type, then every $v_i \in D(\tau)$ is called **complex value**.

Based on this definition of atomic and complex values, a candidate or centroid structure instance can be defined as follows.

Definition 4.4.10 ***Offer or Request Structure Instance (OSI, RSI)***

1. An instance of an offer or request structure $(\psi_1, \ldots, \psi_n)$ (offer or request structure instance (OSI, RSI)) is a record

$$(v_1, \ldots, v_n)$$

   where each v_i is in $D(\psi_i)$.
2. The instance of an OSI is called **offer** (OF_i).
   The instance of an RSI is called **request** (RQ_i).

Now the components are introduced to provide the definition of a structured Grappa matchmaking base. This base is preparing the definition of the Grappa matchmaking system which is one key contribution of this work. The Grappa matchmaking base defines a common basis for matchmaking operations.

Definition 4.4.11 ***Structured* Grappa *Matchmaking Base (SGMB)***

A **Structured** Grappa **Matchmaking Base (SGMB)** is a 7-tuple

$$\Phi := ((\mathcal{T}, TD), \mathcal{M}_d, \mathcal{M}_c, ODS, RDS, MAP, \mathcal{OF})$$

where

1. $(\mathcal{T}, TD)$ is a valid type set.
2. $\mathcal{M}_d$ is a set of distance functions (or metrics) for atomic values. For every $\tau \in \mathcal{B}$ there exists a distance function or metric $d : D(\tau) \times D(\tau) \rightarrow \mathbb{R}^+$.
3. 1. $\mathcal{M}_c$ is a set of combined distance functions each of which is defined for a complex type $\tau \in \mathcal{T} \backslash \mathcal{B}$, which is composed of type(s) $\tau_1, \ldots, \tau_n \in \mathcal{T}$, $n \geq 1$.
   2. Let $v_1, v_2 \in D(\tau)$ be the complex values which are composed of values $v_i^1, \ldots, v_i^n$ for $i = 1, 2$. Then a combined distance function $d : D(\tau) \times D(\tau) \rightarrow \mathbb{R}^+$ is defined on top of the distance functions $d_1, \ldots, d_n$, for the type(s) $\tau_1, \ldots, \tau_n$. For each type $\tau \in \mathcal{T}$ there exists at least one combined distance function $d \in \mathcal{M}_c$. Different combined distance functions are

possible. Formal definitions are given in Section 4.5.4 and Section 4.5.5.
4. *ODS* and *RDS* are the offer and request description schemes $(\psi_1, \ldots, \psi_n)$, where $\psi_i \in \mathcal{T}$ for $i = 1, \ldots, n$. *ODS* and *RDS* are mandatory for all participants of this matchmaking base.
5. *MAP* is the mapping between the offer and the request description scheme. For the scope of this definition it is the identity. A universal definition of *MAP* is provided in Section 4.4.3.
6. $\mathcal{OF}$ is a set $\mathcal{OF} := \{OF_j \mid j = 1, \ldots, l\}$ of instances of ODS, where each $o_i \in \mathcal{OF}$ is an candidate.

Now the Grappa matchmaking system is defined in the following. This system will allow to describe formal semantics of matchmaking computations in the next sections.

Definition 4.4.12 Grappa ***Matchmaking System (GMS)***

A Grappa **Matchmaking System (GMS)** is a pair $(\Phi, \mathcal{A})$ of a SGMB Φ and a matchmaking agent $\mathcal{A}$. Where
1. Φ is a valid SGMB.
2. $\mathcal{A}$ is a matchmaking agent. It can handle the following performatives:
   - *Offer:* Operation which inserts a candidate OF_i into the offer database $\mathcal{OF}$ of the SGMB Φ.
   - *Withdraw:* Operation which removes a candidate OF from the offer database $\mathcal{OF}$ of the SGMB Φ.
   - *Request-k-Neighbor:* Request the k nearest neighbors (candidates in the offer database $\mathcal{OF}$) to a supported centroid RQ.
   - *Request-r-Range:* Request all neighbors (candidates in the offer database $\mathcal{OF}$) which are within the range of r of the supported centroid RQ.

In the next section basic types for multidimensional matchmaking are introduced. These types are applied in the HrNetAgent application and are expandable for any other application.

4.4.2 Basic Types in Grappa

In the following Section 4.4.2 the application of Grappa in practice and the specification of ODS and RDS in XML are introduced. Therefore, a set of basic types is defined. Some complex types are introduced for a specific example. Offers and requests are customized from these types and then transformed to XML.

Basic Types for Multidimensional Matchmaking For the purpose of multidimensional matchmaking some concrete types as shown in table 4.1 must be defined. These allow to generate candidates and centroids.
By applying any valid type definition TD on $\mathcal{B} := \{\tau_1, \ldots, \tau_9\}$ a type set $\mathcal{T}$ is obtained which contains $\mathcal{B}$ as well as complex types living above $\mathcal{B}$. Though there are relationships between several of these types it is useful to define those as basic types. The reason for this is to simplify the process of matchmaking as well as to preserve the special semantics of these types.

Motivations to define compound basic types such as the FreeText type (τ_3) are manifold. Considering free text as a list of keywords (τ_1), which are obtained using a keyword extraction system, only measures obtained from pairwise comparison of the basic values of this list can be regarded. This will not supply a satisfactory matchmaking result. On the other hand, using FreeText as a basic type, the advantage of handling this type using results from information retrieval theory are given. Generally spoken, using the FreeText basic type instead of the complex type FeeText, which would be a set of words, would entail to lose the semantics of words interdependencies (see also Vorhees 1994).

τ_1	`Keyword`	Nouns of a natural language (e.g. English, German, etc.)
τ_2	`Boolean`	True or false
τ_3	`FreeText`	Any combination of words, special characters and numbers
τ_4	`Number`	Integer or real value
τ_5	`Interval`	A array of the type v_1, v_2 where $v_i \in [-maxreal; \ maxreal]$ or $v_i \in [-maxint; \ maxint]$ for $i = 1, 2$
τ_6	`DateInterval`	An array of the type $d.m.y - d.m.y$ where $d \in \{1, \ldots, 31\}$, $m \in \{1, \ldots, 12\}$ and $y \in \{1970, \ldots, maxint\}$
τ_7	`Time`	An array of the type $y : d : h : m : s : m_s$ where $y \in \mathbb{N}$,$d \in \{0, \ldots, 365\}$,$h \in \{0, \ldots, 23\}$, $m \in \{0, \ldots, 59\}$, $s \in \{0, \ldots, 59\}$ and $m_s \in \{0, \ldots, 99\}$
τ_8	`ProperName`	Any proper Name
τ_9	`Geographic`	Address specification and regional preferences

Table 4.1. Basic Types for Concrete Matchmaking in the Grappa ORDL

Example 4.4.13 ***Concrete ODS and RDS in the HR domain***

In this example the atomic types introduced in Table 4.1 are aggregated to complex types. These complex types are elements for an ODS and RDS.

$$\mathcal{B} := \{\tau_1, \tau_2, \tau_3, \tau_5, \tau_6, \tau_7, \tau_8, \tau_9\} = \{\texttt{Keyword}, \texttt{Boolean}, \texttt{FreeText}, \texttt{Number}, \texttt{Interval}, \texttt{DateInterval}, \texttt{Time}, \texttt{ProperName}, \texttt{Geographic}\}$$

Based on these atomic types the following complex types are defined. The complex type `applicantprofile` represents an ODS, the `roleprofile` represents a RDS.

`roleprofile`	:=	(`description`, `professions`, `experiencelevel`, `subjects`, `educationlevel`, `carrequired`, `address`, `generalrequirements`, `specialrequirements`, `languages`, `salary`, `datespecification`, `isintraining`)
`applicantprofile`	:=	(`description`, `professions`, `age`, `car`, `adresspreference`, `desiredjob`, `jobhistory`, `educationhistory`, `generalskills`, `specialskills`, `languageskills`, `salary`, `datespecification`, `isintraining`)
`description`	:=	`FreeText`
`professions`	:=	{`profession`$^+$}
`profession`	:=	`Keyword`
...		
`salary`	:=	(`amount, currency, isnegotiable`)
`amount`	:=	`Number`
`currency`	:=	`Keyword`
`isnegotiable`	:=	`Boolean`
`datespecification`	:=	(`DateInterval, isflexible`)
`dateinterval`	:=	`DateInterval`
`isflexible`	:=	`Boolean`

Specifying ODSs and RDSs in XML The eXtensible Markup Language (XML) describes a class of objects – XML documents – as well as partially the behavior of programs which process these documents. XML is an application profile of the Standard Generalized Markup Language SGML. XML documents are SGML conform. XML-documents are constructed from units called entities which contain either parsed or unparsed data. Parsed data consist of characters, some of which represent character data and other markups. XML

provides a mechanism to specify delimiters of the partition and the markup. A XML-DTD is a document type definition. This is a scheme which describes a class of documents.
XML is designed to specify logically partitioned document classes. This gives the possibility to pre-parse the candidates and centroids for the usage in GRAPPA by applying a XML-parser. The results are structure conform documents which are guaranteed to possess the logical partition which is desired. The type validity must still be parsed at the GRAPPA matchmaker.

Definition 4.4.14 ***Mapping from*** GRAPPA ***ORDL*** $\rightarrow$ ***XML-DTD***

Any basic type $NAME$ is mapped to the data structure `PCDATA` in XML which denotes **parsed character data**. This provides a pre-parsing of the values for GRAPPA. A GRAPPA parser has to be added to be invoked after the successful XML-parsing. If $NAME$ is a basic type, then it is mapped to XML as follows:

$$NAME \rightarrow \texttt{<!ELEMENT } NAME \texttt{ (\#PCDATA)>}$$

If $NAME$ is of set or list type:

$$NAME \rightarrow \texttt{<!ELEMENT } NAME \texttt{ (}\tau\texttt{*)>}$$

where τ is the name of the element the set or list is defined by.

If $NAME$ is of array type of length n:

$$NAME \rightarrow \texttt{<!ELEMENT } NAME \texttt{ (}\tau, \ldots, \tau\texttt{)>}$$

where $(\tau, \ldots, \tau)$ (n entries) is an array of elements of type τ. τ is the name of the element the new element $NAME$ is defined by.

If $NAME$ is of record type:

$$NAME \rightarrow \texttt{<!ELEMENT } NAME \texttt{ (}\tau_1, \ldots, \tau_k\texttt{)>}$$

where $(\tau_1, \ldots, \tau_k)$ is a record of elements (in the right order) the record $NAME$ is defined by. $\tau_1, \ldots, \tau_k$ are the the names of the elements the new element `NAME` is defined by. Additionally the attribute `#REQUIRED` can be added after the content field (`#PCDATA`, etc.). This denotes that this field must be instantiated. Otherwise the instance is not parsed as valid.

Except these definitions basic types $NAME$ where only a choice is offered and no arbitrary inputs are offered can be mapped to an attribute list.

$$NAME \rightarrow \begin{array}{l}\texttt{<!ELEMENT } NAME \texttt{ (\#PCDATA)>} \\ \texttt{<!ATTLIST } NAME \texttt{ VARIABLE (a|b|} \cdots \texttt{|z)>}\end{array}$$

Where $a, \ldots, z$ denote the values or strings the choice can be made from.

```
<!ELEMENT DOCTYPE (roleprofile)>
<!ELEMENT roleprofile (description, professions,
    experiencelevel, subjects, educationlevel, carrequired,
    address, generalrequirements, specialrequirements,
    languages, salary, datespecification, isintraning)>
    <!ELEMENT description (#PCDATA)>
    <!ELEMENT professions (profession*)>
        <!ELEMENT profession (#PCDATA)>
    <!ELEMENT experiencelevel (#PCDATA)>
    <!ELEMENT generalrequirements (skill*)>
    ...
    <!ELEMENT salary (amout,currency,isnegotiable)>
        <!ELEMENT amount (#PCDATA)>
        <!ELEMENT currency (#PCDATA)>
        <!ATTLIST isngegotiable (true|false)>
    <!ELEMENT datespecification (dateinterval,isflexible)>
        <!ELEMENT dateinterval (#PCDATA)>
        <!ATTLIST isflexible (true|false)>
```

Table 4.2. Sample XML-DTD Using Grappa

In the following example it is illustrated how XML is applied in Grappa. This is shown by mapping the ODS and RDS to a XML-DTD.

Example 4.4.15

The ODS and RDS shown in example 4.4.13 are transformed into XML-DTDs. In Table 4.2 the XML-DTD for the RDS is shown. A XML-DTD describes a class of documents. This class can be instantiated. Table 4.3 shows the ODS instance of a computer scientist looking for a vacant position. The OSI advertises a computer scientist who is looking for a job in the near future until the end of 2005. Table 4.4 shows the RSI from a company who is looking for a computer scientist.

In the following example a simplified multidimensional matchmaking case is shown. The full introduction to the use case HrNetAgent is introduced in detail in Chapter 5 and 6. In the next example simplified instances from the HrNetAgent prototype are provided.

Example 4.4.16 *Simplified* HrNetAgent

Candidates and centroids in the HrNetAgent example provide a multidimensional structure (ODS and RDS, respectively). Instances of those objects, OSIs and RSI's are provided in Example 4.4.15. Due to the fact that the ODS and RDS schemes are heterogeneous a mapping is needed which provides

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<applicantprofile>
    <description>I have experience designing and
implementing
        complex applications. For several years I have been
        working in a team implementing large scale
prototypes
        based on agent systems and distributed systems.
        Most clients came from the banking and commodity
        exchange domain.
    </description>
    <professions>
        <profession>Computer Scientist</profession>
    </professions>
    <generalskills>
        <skill>
            <attribute>Soft Skills</attribute>
            <value>5</value>
        </skill>
        <skill>
            <attribute>Team player</attribute>
            <value>5</value>
        </skill>
    </generalskills>
    ...
    <salary>
        <amount>65000</amount>
        <currency>EUR</currency>
        <flag>5</flag>
        <isnegotiable>true</isnegotiable>
    </salary>
    <datespecification>
        <dateinterval>Aug 30 00:00:00 GMT+01:00 2002 ;
        Dec 31 00:00:00 GMT+01:00 2005</dateinterval>
        <isflexible>true</isflexible>
    </datespecification>
</applicantprofile>
```

Table 4.3. Sample OSI as XML-Instance

- an assignment of the attributes and dimensions from the ODS to the RDS.
- an assignment of the attribute and dimension pairs towards *atomic* and *complex distance functions.*

In the following section a generic mapping among heterogeneous ODS and RDS schemes is introduced.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<roleprofile>
    <description>We are looking for a mathematician or computer
        scientist to enlarge our team. The focus of our
        team is the design, implementation and evaluation of
        software prototypes based on software agent technology.
        The prototypes are developed for banks, trade
        and commodity exchange companies.
    </description>
    <professions>
        <profession>Computer Scientist</profession>
        <profession>Mathematician</profession>
    </professions>
    <experiencelevel>4</experiencelevel>
    ...
    <generalrequirements>
        <skill>
            <attribute>Team player</attribute>
            <value>5</value>
        </skill>
        <skill>
            <attribute>Soft Skills</attribute>
            <value>3</value>
    </skill>
    </generalrequirements>
    <salary>
        <amount>50000</amount>
        <currency>EUR</currency>
        <isnegotiable>true</isnegotiable>
    </salary>
    <datespecification>
        <dateinterval>Oct 30 00:00:00 GMT+01:00 2002 ;
        Dec 31 00:00:00 GMT+01:00 2003</dateinterval>
        <isflexible>true</isflexible>
    </datespecification>
    <istraining>false</istraining>
</roleprofile>
```

Table 4.4. Sample RSI as XML-Instance

4.4.3 Heterogeneous Offer and Request Profiles

As introduced in the previous section, ODS and RDS are possibly complex offer and request schemes. These schemes do not have to be of the same structure. RDS and ODS can vary in attribues or even dimensions. The idea is now to define a mapping which allows automated matchmaking on heterogeneous offer and request structures. For this purpose the mapping MAP is introduced which is defined in this section and illustrated in an example which is given in Figure 4.2.

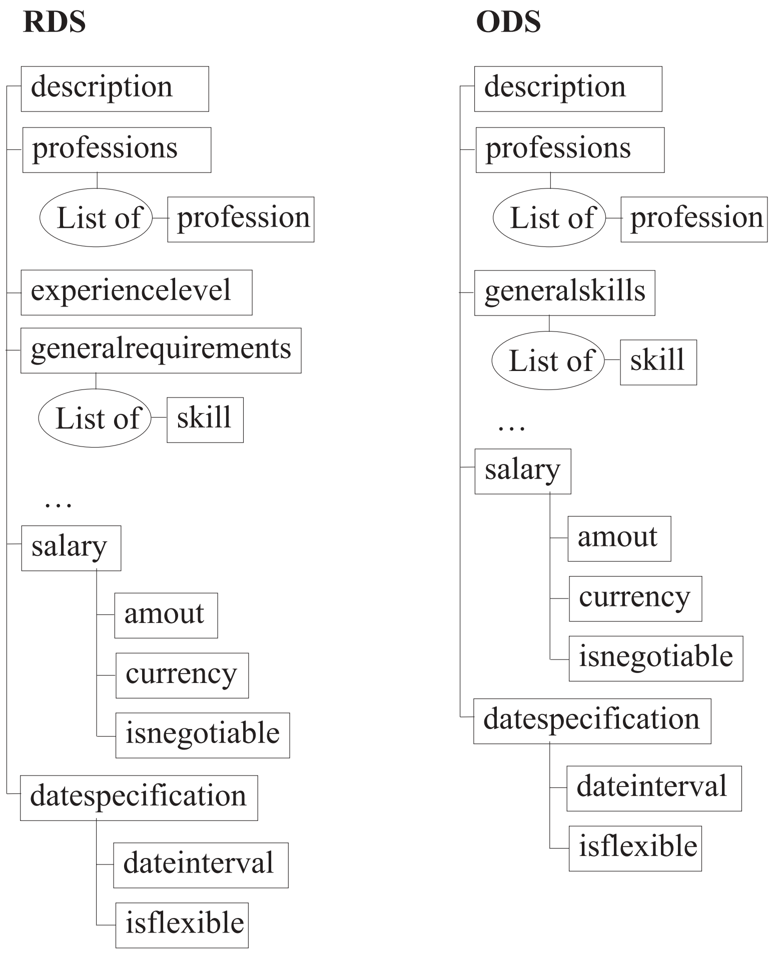


Figure 4.2. Example RDS and ODS

Definition 4.4.17 ***Mapping MAP***

MAP defines a mapping which assigns dimensions and attributes among heterogeneous centroid and candidate structures. To each dimension or attribute $\varphi_i, \psi_i \in \mathcal{T}$ (for all i) of the scheme RDS= $(\psi_1, \ldots, \psi_m)$ a function is assigned

$$MAP(\psi_i) : D(ODS) \to D(\psi_i),$$

such that $MAP(\psi_i) = f_{\psi_i}(\varphi_1, \ldots, \varphi_k)$, where f_{ψ_i} is a function on functions and $\varphi_1, \ldots, \varphi_k$ are subschemes of ODS, viewed as deconstructor functions on the instances of ODS. $MAP(\psi_i)(o)$ constructs for the request component ψ_i a value, given any candidate o. Thus, a request object $MAP(o) = (MAP(\psi_1)(o), \ldots, MAP(\psi_m)(o))$ can be constructed from o, which can be used for computing the distance between o and a given centroid r. Any non-record type RDS is viewed as record type (RDS) and its MAP is defined respecitvely.

To the knowledge of the author, in all other state-of-the-art matchmaking systems from which the most important ones are introduced in Section 3.1, ODS and RDS coincide, and MAP is identity, i.e., $MAP(\psi_i) = \psi_i$. In most real life scenarios the offer and request structures are not identical. An application example where ODS and RDS differ, taken from the HrNetAgent application, is shown in Figure 4.2. There, MAP is identity for several components (e.g. , $MAP($`description`$) =$ `description`, where `description` is used to name the component of this type), while it assigns to `Profession`$^+$ the list obtained by concatenating `professions` and `generalskills`.

Special cases are that RDS is a subscheme of ODS and vice versa. Here MAP is straightforward: in the former case, it projects out components of ODS, while in the latter, $f_{\psi_i}(\varphi_1, \ldots, \varphi_k)$ may add missing attributes to a candidate and assign dummy values to them. In practice, f_{ψ_i} may perform various complex operations such as merging lists, taking the union of sets, combining values into a complex value (e.g. , assemble dates) etc.

In the next section distance functions and metrics are defined. Here, the mathematical basis is layed by supporting basic definitions for metrics and distance functions.

4.5 Distance Functions and Metrics

In this section distance functions and metrics are focussed. In the last sections the existence of distance functions was postulated. Some of the distance functions, which are computing distances among basic and complex values, i.e. attribute instances, are defined for the entire framework. To cover domain semantics other application specific distance functions are defined.

In Section 4.5.1 the properties and definitions for metrics and distance functions are introduced. In Section 4.5.2 distance function approaches for free texts are defined. The definition of these distance functions is an important contribution of this work. In the following Section 4.5.3 basic distance functions for numbers, intervals, dates and some more basic types are introduced. In Section 4.5.4 complex distance functions are introduced. Here, the weighted average distance as well as the Hausdorff and the minimum-link distance are playing key roles. The chapter is wrapped up by Section 4.5.5, where weighting paradigms for complex types are described.

4.5.1 Metric Definition and Properties

The aim of this section is to present formal semantics for the following kind of query to a GMS:
"What are the closest $k \geq 0$ candidates in $\mathcal{OF}$ matching my centroid RQ?
and
"Show me all candidates in $\mathcal{OF}$ which are within the range of r to my centroid RQ!"
In order to process queries like this, distance measures among basic and complex types are needed. These distance measures are used to compare every incoming centroid RQ with all candidates stored in $\mathcal{OF}$.
Let $\mathcal{T}$ be a valid type set. This set consists of basic types $\mathcal{B}$ as well as a set of complex types $\mathcal{T}\backslash\mathcal{B}$ which are defined using a set of valid type definitions TD. Now the existence of a distance function for each type $\tau \in \mathcal{B}$ is postulated. Concrete distance functions for the types are defined in the next sections.

Definition 4.5.18 ***Distance Function, Metric***

Let $\tau_i \in \mathcal{B}$ be any basic type. Let further $D(\tau_i)$ be the domain of τ_i. Let $d_i : D(\tau_i) \times D(\tau_i) \to \mathbb{R}^+$ be a function on the domain $D(\tau_i)$ mapping to $\mathbb{R}^+$. If the properties
(i) $d_i(v_1, v_2) = 0 \Longleftrightarrow v_1 = v_2$ for all $v_1, v_2 \in D(\tau_i)$.
(ii) $d_i(v_1, v_2) = d(v_2, v_1)$ for all $v_1, v_2 \in D(\tau_i)$.
(iii) $d_i(v_1, v_3) \leq d_i(v_1, v_2) + d_i(v_2, v_3)$ for all $v_1, v_2, v_3 \in D(\tau_i)$.
are fulfilled, d_i is called **metric** on $D(\tau_i)$. If only (i) and (ii) hold, d_i is called **distance function** on $D(\tau_i)$.

In order to provide a meaningful complex matchmaking the individual distance measures among atomic values must be of the same order. If an average or a weighted average between values of very different order is computed no satisfactory matchmaking result will be obtained. For this reason on top of an existing metric the following metric is defined.

Theorem 4.5.19

Let d be a metric on the domain of type τ – $D(\tau)$ with $d : D(\tau) \times D(\tau) \to \mathbb{R}^+$. Then $\tilde{d} : D(\tau) \times D(\tau) \to [0, 1)$ where

$$\tilde{d}(v_1, v_2) := \frac{d(v_1, v_2)}{1 + d(v_1, v_2)}$$

with $v_1, v_2 \in D(\tau)$ is a metric on $D(\tau)$, too.

Proof: It must be shown that (i),(ii) and (iii) of definition 4.5.18 hold. It is obvious that $\tilde{d} : D(\tau) \times D(\tau) \to \mathbb{R}^+$ is a mapping.

(i) $\tilde{d}(v_1, v_2) = \frac{d(v_1,v_2)}{1+d(v_1,v_2)} \geq 0$ for all $v_1, v_2 \in D(\tau)$.

(ii) $\tilde{d}(v_1, v_2) = 0 \Leftrightarrow d(v_1, v_2) = 0 \Leftrightarrow v_1 = v_2$

(iii) Let $v_1, v_2, v_3 \in D(\tau)$. Then

$$\begin{aligned}
\tilde{d}(v_1, v_3) &= \frac{d(v_1, v_3)}{1 + d(v_1, v_3)} = \frac{1}{1 + (d(v_1, v_3))^{-1}} \\
&\underset{*}{\leq} \frac{1}{1 + (d(v_1, v_2) + d(v_2, v_3))^{-1}} = \frac{d(v_1, v_2) + d(v_2, v_3)}{1 + d(v_1, v_2) + d(v_2, v_3)} \\
&= \frac{d(v_1, v_2)}{1 + d(v_1, v_2) + d(v_2, v_3)} + \frac{d(v_2, v_3)}{1 + d(v_1, v_2) + d(v_2, v_3)} \\
&\underset{**}{\leq} \frac{d(v_1, v_2)}{1 + d(v_1, v_2)} + \frac{d(v_2, v_3)}{1 + d(v_2, v_3)} \\
&= \tilde{d}(v_1, v_2) + \tilde{d}(v_2, v_3)
\end{aligned}$$

$*$: because d is a metric on $D(\tau)$.
$**$: because $d(v_1, v_2) \geq 0$ and $d(v_2, v_3) \geq 0$.

q.e.d.

This new metric provides a metric which can be applied on all existing metrics and distance functions. Under certain circumstances it is still not advisable to apply this metric. This is, e.g. , the case if the distribution of the results of the original metric d is too close to 0.

For each basic type τ a distance function must be defined. This distance function can be very different from type to type. For the basic types introduced in Section 4.4.2 domain specific distance functions are provided in the Sections 4.5.2 and 4.5.3. First of all the existence of a distance function for each type is postulated.

Definition 4.5.20 *Type Compatibility*

Let $\tau_1 \in \mathcal{T}$ and $\tau_2 \in \mathcal{T}$ be two types. Let $\sim$ denote the compatibility of two types. Then

$$\tau_1 \sim \tau_2 \Leftrightarrow \begin{cases} \text{if } \tau_1, \tau_2 \in \mathcal{B} \text{ and } \tau_1 \text{ is identical to } \tau_2 \\ \text{if } \tau_1, \tau_2 \in \mathcal{T} \backslash \mathcal{B} \text{ and } \tau_1 \text{ and } \tau_2 \text{ are defined applying} \\ \quad \text{the same type constructors on the same subset of types.} \end{cases}$$

Postulate 4.5.21 *Comparison of Types*

1. The distance between two values $v_1 \in D(\tau_1)$ and $v_2 \in D(\tau_2)$ can be measured if and only if type τ_1 and τ_2 are compatible. Then the values v_1 and v_2 are called **comparable**.

2. For each basic type $\tau_i \in \mathcal{B}$, $i = 1, \dots, m$ there exists at least one **distance function** d which operates over the domain $D(\tau_i)$ of τ_i.
   This distance function $d_i : D(\tau_i) \times D(\tau_i) \to \mathbb{R}^+$ computes the distance between two atomic values of type τ_i.

Now a distance measure can be defined. This measure is introduced to compute the distance between two atomic values of the same type τ.

Definition 4.5.22 ***Distance Measure***

> Let $\tau \in \mathcal{T}$ be an atomic type. Let v_r, v_j be atomic values in $D(\tau)$. Let v_r be taken from a centroid RQ and v_j be taken from a stored candidate $OF_j \in \mathcal{OF}$. The measure $\delta_j = d(v_r, v_j) \in \mathbb{R}^+$ computed by the distance function of type τ is called **distance measure with respect to** RQ between the value $v_r \in D(\tau)$ and the value $v_j \in D(\tau)$. This distance measure is computed by the distance function d of τ.

The mathematical properties of metrics and distance functions are provided in the next two sections and functions are introduced which partly apply these properties for the definition of domain specific matchmaking functions.

4.5.2 Distance Functions for the Basic Type FreeText

In this section the most important distance function in the GRAPPA framework is introduced. This distance function is derived from the research domain of Information Retrieval (IR). Salton (1989) has contributed essentially to this field and provides the basis for the ideas which are formally defined in the following. Important details which have been introduced and discussed by Salton and his group are also published in Salton (1988), Salton et al. (1993) and Vorhees (1994).

Let $d_i \in D(\tau)$ be a document. Let further $t_1, \dots, t_k$ be the terms (without the stop words, e.g. , "and", "or", "we", ...) occurring in document d_i. Let $D(\tau)$ consist of N documents. Let $t_j \in \{t_1, \dots, t_k\}$ be a term which occurrs in document d_i. Then the **document frequency**

$$df_j \in \mathbb{N} \text{ and } df_j \leq N$$

is the number of documents in which term t_j occurs at least once and the **term frequency**

$$tf_{ij} \in \mathbb{N}$$

is the frequency of term t_j in document d_i.

Therefore a term discrimination value can now be defined as the inverse document frequency factor. This factor has the property to discriminate between terms which have a high relevance within some documents but do have little relevance in the documents of the complete collection.

Definition 4.5.23 ***Inverse Document Frequency Factor***

Let $1 \leq N \in \mathbb{N}$ be the number of documents in $D(\tau)$. The **inverse document frequency** factor for term t_j is defined as

$$idf_j := \log \frac{N}{df_j}$$

Theorem 4.5.24

Let $\tau :=$ FreeText be a basic type. Let $D(\tau)$ be a collection of documents of type FreeText. Let $1 \leq N \in \mathbb{N}$ be the number of documents in $D(\tau)$. Let T be the set of terms which are extracted from all documents in $D(\tau)$. Then for every term $t_i \in T$

$$0 \leq idf_i \leq \log N$$

Proof: Let $t_i \in T$ be any term extracted from any document in $D(\tau)$. Then $df_i \geq 1$ because if term t_i is in T it must at least occur in one document of the collection.
On the other hand $df_i \leq N$ because the term t_i can at the most occur in each of the N documents of the collection. So

$$1 \leq df_j \leq N.$$

$\Longrightarrow 1 \leq \frac{N}{df_j} \leq N$ and finally $\Longrightarrow 0 \leq \log \frac{N}{df_j} \leq \log N$. *q.e.d.*

The inverse document frequency factor indicates the importance of a term within the collection of N documents. This importance factor reflects the strength of term t_j with respect to the collection $D(\tau)$. If this factor is combined together with the term frequency factor to a weight for a term t_j which is in a collection $D(\tau)$ a strong indicator for distance computations among an incoming centroid q and each document within the collection is obtained.

Document collection $D(\tau)$ revistited. Let T be all terms extracted from the documents in $D(\tau)$. Let $n \in \mathbb{N}$ be the number of terms in T. $k \leq n$ terms are distinguished by computing the inverse document frequency factor idf_j for each term $t_j \in T$ and then select the top $k \leq n$ terms as indexing terms for which $idf_j > 0$ is valid.

A document space is invoked, represented by $k \leq n$ terms out of T. A basis for this document space is $B := \{v_1, \dots, v_k\}$ where

$$v_j := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \leftarrow \text{Position } j$$

Term	df_j	$w_{1,j}$	$w_{2,j}$	Term	df_j	$w_{1,j}$	$w_{2,j}$
agent	10	0,48	0,48	focus	3	1,00	0,00
application	5	0,00	0,78	implement	19	0,20	0,40
bank	2	1,18	1,18	large	5	0,00	0,78
base	6	0,70	0,70	look	6	0,70	0,00
client	5	0,00	0,78	mathematician	6	0,70	0,00
commodity	1	0,00	1,48	prototype	4	1,75	0,88
complex	4	0,00	0,88	scale	3	0,00	1,00
computer	10	0,48	0,00	scientist	1	1,48	0,00
design	30	-	-	several	2	0,00	1,18
develop	10	0,48	0,00	software	25	0,16	0,00
distribute	6	0,00	0,70	system	21	0,00	0,31
domain	12	0,00	0,40	team	20	0,35	0,18
enlarge	4	0,88	0,00	technology	5	0,78	0,00
evaluation	5	0,78	0,00	trade	3	1,00	0,00
exchange	2	0,00	1,18	working	4	0,00	0,88
experience	15	0,00	0,30	years	8	0,00	0,57

Table 4.5. Document Frequencies and Weights in the Example Text.

Now each document $d_i \in D(\tau)$ can be represented by a vector DV_i which is a linear combination of the vectors $v_i \in B$. Based on the inverse document frequency factor and the term frequency factor a weight for each term t_j within document d_i can be defined.

$$w_{i,j} := tf_{i,j} \log \frac{N}{df_j} = tf_{i,j} idf_j$$

Then the document vector for document d_i has the form:

$$\begin{aligned} DV_i &= (w_{i,1}, \dots, w_{i,k})' \\ &= (tf_{i,1} \cdot idf_1, \dots, tf_{i,k} \cdot idf_k)' \\ &= (1, 0, \dots, 0)' \cdot idf_1 tf_{i,1} + (0, 1, 0, \dots, 0)' \cdot idf_2 tf_{i,2} + \dots \\ &\quad + (0, \dots, 0, 1, 0)' \cdot idf_{k-1} tf_{i,k-1} + (0, \dots, 0, 1)' \cdot idf_k tf_{i,k} \\ &= v_1 \cdot idf_1 tf_{i,1} + \dots + v_k \cdot idf_k tf_{i,k} \end{aligned}$$

Example 4.5.25

Example 4.4.15 continued. Here, a document collection of 30 documents where the terms have the occurrence pattern shown in table 4.5. The weights shown in this table are computed using the above introduced inverse document frequency factor formula. The weights for the term *design* are not computed due to the fact that a term which occurs in all documents of a collection does not provide any discriminating power.

The set of all document vectors DV_i, which are extracted from the documents in the candidate database is called document space, denoted by Ω_d.
Now the distance between two document vectors DV_i and DV_l in a document space Ω_d based on the Euclidean distance function can be defined as follows.

Definition 4.5.26

Let DV_i and DV_l be two document vectors in a document space. Then the mapping $d : \Omega_d \times \Omega_d \to \mathbb{R}^+$

$$d(DV_i, DV_l) := \|DV_i - DV_l\|_2 = \sqrt{\sum_{j=1}^{k}(w_{i,j} - w_{l,j})^2}$$

is defined as the distance between document vector DV_i and DV_l.

Theorem 4.5.27

Let Ω_d be a document space. Let $DV_i \in \Omega_d$ and $DV_l \in \Omega_d$ then

$$d(DV_i, DV_l) = \sqrt{\sum_{j=1}^{k}(w_{i,j} - w_{l,j})^2}$$

is a distance function.

Proof: It has to be shown that (i) and (ii) of definition 4.4.18 hold.

(i) ($\Rightarrow$) Let $DV_i, DV_l \in \Omega_d$. Let now $d(DV_i, DV_l) = 0$. Suppose that $DV_i \neq DV_j$. Then at least one entry is different $w_{i,j} \neq w_{l,j}$ for at least one $j \in \{1, \ldots, k\}$.

$$\begin{aligned} & d(DV_i, DV_l) = 0 \wedge w_{i,j} \neq w_{l,j} \text{ for at least one } j \in \{1, \ldots, k\} \\ \Leftrightarrow \quad & \sqrt{\sum_{j=0}^{k}(w_{i,j} - w_{l,j})^2} = 0 \wedge w_{i,j} \neq w_{l,j} \\ \Leftrightarrow \quad & \sum_{j=0}^{k}(w_{i,j} - w_{l,j})^2 = 0 \wedge w_{i,j} \neq w_{l,j} \end{aligned}$$

Each summand satisfies $(w_{i,j} - w_{l,j})^2 \geq 0$. And together with the assumption at least one summand $(w_{i,j} - w_{l,j})^2 > 0$ due to the fact that at least one $w_{i,j} \neq w_{l,j}$.

$$\Leftrightarrow \qquad \sum_{j=0}^{k}(w_{i,j} - w_{l,j})^2 > 0$$

This of course contradicts the assumption that $d(DV_i, DV_l) = 0$.
($\Leftarrow$) Let $DV_i = DV_l$. This means that $w_{i,j} = w_{l,j}$ for all $j \in \{1, \dots, k\}$. From this fact directly follows that

$$d(DV_i, DV_l) = \sqrt{\sum_{j=0}^{k}(w_{i,j} - w_{l,j})^2} = 0$$

This proves (i) of definition 4.4.18.

(ii) Let $DV_i, DV_l \in \Omega_d$.

$$d(DV_i, DV_l) = \sqrt{\sum_{j=0}^{k}(w_{i,j} - w_{l,j})^2} = \sqrt{\sum_{j=0}^{k}(w_{l,j} - w_{i,j})^2} = d(DV_l, DV_i)$$

This proves (ii) of definition 6.1.

Alternatively the distance between two documents in the document space can be defined using the cosine between the two document vectors.

Definition 4.5.28 ***Cosine Distance Measure***

Let $DV_i, DV_l \in \Omega_d$ be two document vectors. Then the distance measure between these document vectors is defined as

$$d_s(DV_i, DV_l) := 1 - \frac{\sum_{j=1}^{k} w_{i,j} \cdot w_{l,j}}{\sqrt{\sum_{j=1}^{k}(w_{i,j})^2 \sum_{j=1}^{k}(w_{l,j})^2}}$$

This equation represents one minus the cosine of the angle between the document vectors DV_i and DV_l considered as vectors in our k dimensional document space.

This definition has several advantages in term weighting. For example if two document vectors are compared using this distance measure d_s a term which does not occur in one of the documents does not affect the summarized distance at all. On the other hand the disadvantage of this distance measure is that it has not even the properties of a distance function.

Example 4.5.29

Example 4.4.25 continued. The distance between document d_1 and d_2 can now be computed as the Euclidean or cosine distance between the corresponding document vectors DV_1 and DV_2 which can be seen in table 4.5 in the $w_{1,j}$ and $w_{2,j}$ columns, respectively. The Euclidean distance is:

$$d(DV_1, DV_2) = \sqrt{\sum_{j=1}^{32}(w_{1,j} - w_{2,j})^2} = 4,35$$

Where the cosine distance measures:

$$d_s(DV_1, DV_2) := 1 - \frac{\sum_{j=1}^{32} w_{1,j} \cdot w_{2,j}}{\sqrt{\sum_{j=1}^{32}(w_{1,j})^2 \sum_{j=1}^{3} 2(w_{2,j})^2}} = 0,71$$

Here, the cosine distance measure reflects a distance of 71 percent i.e. a relevance of 29 percent. The Euclidean distance is a measure which can only be judged in relation to other distances from the same example domain. Then a mapping to the Interval $[0, 1)$ would apply and a distance value – measured in percent – would be the result.

Space Density Considerations Besides the inverse document frequency factor there are different other considerations to distinguish documents from each other. One aspect are considerations about the density of the document space.

Here, a canonical base for the document space $B := \{v_1, \ldots, v_n\}$ as defined above is considered. A weighting system where the weights of a document vector are entirely constructed applying the term frequency factor $tf_{i,j}$ is applied. Based on this simple weight definition where $w_{i,j} := tf_{i,j}$ a preliminary document space Ω'_d can be defined.

The distance function for document vectors $d : \Omega'_d \times \Omega'_d \rightarrow [0, 1]$ defined like the distance function in the last section can still be applied.

To introduce a weighting paradigm based on space density considerations firstly a density notion is defined.

Definition 4.5.30 ***Document Space Density***

Let $DV_i, DV_l \in \Omega_d$ be two document vectors. Let $DV_i := (w_{i,1}, \ldots, w_{i,k})'$ and $DV_l := (w_{l,1}, \ldots, w_{l,k})'$. Let $d : \Omega_d \times \Omega_d \rightarrow [0, 1]$ be the distance function for document vectors defined above. Then

$$\Phi := \frac{\sum_{i=1}^{N} \sum_{l=1, l \neq i}^{N} d(DV_i, DV_l)}{N(N-1)}$$

is the **density** of document space Ω_d.

Now, the difference in density of the document space Ω_d is considered, before and after the assignment of a term t_j as indexing term.

Definition 4.5.31 ***Term Discrimination Value***

Let Φ be the density of document space Ω_d. Let Φ_j be the density of the document space Ω_d, leaving out term t_j for indexing purposes. This reduces the dimension of the document space on $k-1$. Then the **term discrimination value** dv_j for term t_j is defined as

$$dv_j := \Phi_j - \Phi.$$

The term discrimination value is usually

- *Positive* at medium frequency terms (that neither appear too often nor too rarely).
- *Negative* at high frequency terms.
- *Close to Zero* at very low frequency terms.

High frequency and very low frequency terms are not very useful for indexing purposes. The reasons are the following:

1. High frequency terms which are randomly scattered over all documents of the collection do not discriminate the documents from each other and are for this reason extraneous for indexing purposes.
2. Very low frequency terms which only occur in one or two documents of the collection are as well extraneous for indexing purposes because they do not affect the distance between the large part of documents of the collection.

For this reason it is recommended to apply either of the following procedures.

Plain Term Discrimination The first procedure which is in practice more relevant is described here. The second procedure is introduced in the next subsection.

Let $\mathbb{R} \ni \varepsilon > 0$ be a small threshold. Select all terms which have a discrimination value $dv_j > \varepsilon$ and assign them as indexing terms. Let $n \in \mathbb{N}$ be terms extracted from the documents of the collection. Let $\mathbb{N} \ni k \leq n$ be the number of terms which possess a discrimination value $dv_j > \varepsilon$. Then k indexing terms are obtained. The weights can then be defined as

$$w_{i,j} := tf_{i,j} dv_j.$$

Because $tf_{i,j} \geq 0$ and $dv_j > 0$ for all $i \in \{1, \ldots, N\}$ and $j \in \{1, \ldots, k\}$ all weights $w_{i,j} \geq 0$ is valid. Now the Euclidean distance function d from the last section can be applied on the set of weights $\{w_{i,j} \mid i \in \{1, \ldots, N\}, j \in \{1, \ldots, k\}\}$. Due to the fact that all weights are greater or equal to zero the properties of a distance function can again be proved.

Phrase-Formation and Thesaurus Transformation The second possibility using the term discrimination value is the transformation of low frequency and high frequency terms into medium frequency units.

Let $\mathbb{R} \ni \varepsilon > 0$ where ε is small. Let further $I := \{t_1, \ldots, t_{k-1}\}$ be the partition on the set of extracted terms where with $dv_j > \varepsilon$, $H := \{t_k, \ldots, t_l\}$ are the terms where $dv_j < -\varepsilon$ and $L := \{t_{l+1}, \ldots, t_n\}$ where $-\varepsilon \leq dv_j \leq \varepsilon$ (the terms $t_1, \ldots, t_n$ are extracted in arbitrary order and sorted for this partition).

To reduce the frequency of high frequency terms the **phrase formation** is used. This technique bases on the following idea.

Definition 4.5.32 ***Co-occurrence Relationship***

Let $t_1, t_2 \in T$ be terms. Then the **Co-occurrence relationship**

$$cr(t_1, t_2) \in \mathbb{N}$$

between t_1 and t_2 is the number of documents in Ω_d in which t_1 and t_2 both have a term frequency $tf_{i,1} > 0$ and $tf_{i,2} > 0$ respectively.

Now the phrase formation can be defined.

Definition 4.5.33 ***Phrase-Formation***

Let $t_j \in H$ be a high frequency term. Let $th \in \mathbb{N}$ be a certain threshold. Let $PH_j := \{t_i \mid i = 1, \ldots, m\}$ be a set of terms which have Co-occurrence relationships $cr(t_j, t_i) > 0$ for all $i = 1, \ldots, m$ to t_j but all have a document frequency $df_i < th$ for $i = 1, \ldots, m$. Then a **phrase** is an expression which consists of a phrase head t_j and phrase components $t_1, \ldots, t_l \in PH$ where $m \geq l \in \mathbb{N}$ (enumeration may be changed)

$$t_j : t_1 : \cdots : t_l$$

Term t_j is phrase-formatted if term t_j is taken out of the set of indexing terms I and replaced with the term phrase $t_j : t_1 : \cdots : t_l$ which has a higher frequency than t_j.

The method to enlarge the frequency of low frequency terms is called thesaurus transformation. This technique is based on the idea to replace low frequency terms by a unique representative of the class of terms this low frequency term is located in. Therefore a thesaurus is contacted which maintains classes of terms.

Definition 4.5.34 ***Thesaurus***

A **thesaurus** TH is a set of term classes where for each term t_j there exists a class of terms $CL \subset TH$ for which $t_j \in CL$ is valid. Each class CL has a unique representative $t_i \in CL$ which is the head of the class.

Definition 4.5.35 ***Thesaurus Transformation***

> Let $\{t_l \in L \mid l = 1, \ldots, m\}$ be a subset of low frequency terms. Let CL be the thesaurus term class to which $\{t_l \mid l = 1, \ldots, m\}$ belong to. Let $t_i \in CL$ be the unique representative for class CL. Then the process of replacing $\{t_l \mid l = 1, \ldots, m\}$ by t_i as indexing term is called **thesaurus transformation**.

Example 4.5.36

> Example 4.4.25 revisited. The terms `system` and `technology` have the weights 0,00 and 0,31 as well as 0,78 and 0,00, respectively. These terms do not help comparing the documents d_1 and d_2 because they are not in their intersection of terms. If the terms are joined to a thesaurus class CL and one of those is chosen as representative, this class receives the weights 0,08 and 0,16. Now the new term (class) helps comparing the documents.

Many ambiguous phrases can appear when automatic phrase-formation is applied. Additionally this method is very time complex. For this reason it is little relevant for real time indexing. If the thesaurus transformation is used also certain additional problems appear.

The classification of words which have a similar semantic – which are the content of a thesaurus class – is domain specific. E.g., a software agent is not related with software as long as the secret service domain is considered. In the computer science domain *software agent* has definitely a relation with software. So it is obvious that many semantical problems arise when dealing with automatic thesaurus transformation and phrase formation.

In practice these issues are not recommended for real time indexing purposes. Real time indexing is entirely based on a term frequency – inverse document frequency algorithm.

4.5.3 Distance Computations for Numbers, Intervals, Dates, and Times

In this subsection distance functions for numbers, intervals, DateIntervals and time specifications are presented.

Distance Computations for Numbers In the following the distance problem among numbers is considered.

Definition 4.5.37 ***Distance between Numbers***

> Let $v_1, v_2 \in \mathbb{R}$ be two real values. Then the distance function $d : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}^+$ is defined as

$$d_n(v_1, v_2) :=\mid v_1 - v_2 \mid .$$

It is obvious that this distance function even fulfills the properties of a metric.

Distance Function for Intervals The distance problem among intervals is not as trivial as the distance problem among numbers.

Definition 4.5.38 ***Interval***

Let $l, r \in \mathbb{R}$ be two real values where $l < r$. Then an **interval** is an array of length two of real values

$$[l, r].$$

The basic idea to define a distance function for intervals is to measure the amount two intervals are overlapping.

Definition 4.5.39 ***Distance between Intervals***

Let $l_1, l_2 \in \mathbb{R}$ be real values. Let $r_1 \in \mathbb{R}$ and $r_2 \in \mathbb{R}$ be real values so that $i_1 := [l_1, r_1]$ and $i_2 := [l_2, r_2]$ are intervals. Then the distance between the intervals i_1 and i_2 $d : (\mathbb{R} \times \mathbb{R}) \times (\mathbb{R} \times \mathbb{R}) \to \mathbb{R}^+$ is defined as

$$d(i_1, i_2) := \begin{cases} \frac{|l_1-l_2|+|r_1-r_2|}{|l_2-r_1|} & \text{for } l_1 < l_2 < r_1 < r_2 \\ \frac{|l_2-l_1|+|r_2-r_1|}{|l_1-r_2|} & \text{for } l_2 < l_1 < r_2 < r_1 \\ \frac{|l_1-l_2|+|r_2-r_1|}{|l_2-r_2|} & \text{for } l_1 \leq l_2 < r_2 \leq r_1 \\ \frac{|l_2-l_1|+|r_1-r_2|}{|l_1-r_1|} & \text{for } l_2 < l_1 < r_1 < r_2 \\ \infty & \text{otherwise.} \end{cases}$$

It is obvious that this function is a distance function. Due to the fact that the distance between two intervals which are disjoint is ∞ the properties of a metric are not provable. In practice ∞ is changed to a certain maximum value.

Distance Function for DateIntervals Considering DateIntervals the distance problem is fairly easy to solve. Due to the fact that the dates which are allowed in this datatype begin with the 01.01.1970 at 00:00:00 h simply a mapping $t : D(\tau_6) \to \mathbb{N} \times \mathbb{N}$ can be constructed which preserves the order of dates. Let t be a function which maps a date d to the number of seconds which have passed between 01.01.1970 at 00:00:00 h and date d 00:00:00 h.

Due to the fact that $\mathbb{N} \subset \mathbb{R}$ the distance function specified in 4.5.3 can be applied as a distance function for DateIntervals.

Let $di_1, di_2 \in D(\tau_6)$ be two date intervals. Then the distance between di_1 and di_2 is defined as

$$d_{di}(di_1, di_2) := d_i(t(di_1), t(di_2))$$

Where d_{di} is the distance function for DateIntervals to be defined and d_i is the distance function for intervals defined in subsection 4.5.3.

Example 4.5.40

Example 4.4.36 continued. Let $di_1 := [08/30/2002, 12/31/2005]$ and $di_2 := [10/30/2002, 12/31/2003]$ be two date intervals. These intervals are mapped to $\mathbb{N} \times \mathbb{N}$ using the mapping t.

$$t(di_1) = [60991480800000, 61096806000000]$$
$$t(di_2) = [60996754800000, 61033647600000]$$

Now the distance can be computed using the distance function for intervals specified above.

$$\begin{aligned} d(t(d_1), t(d_2)) &= d(t([08/30/2002, 12/31/2005]), t([10/30/2002, 12/31/2003]) \\ &= d([60991480800000, 61096806000000], \\ &\quad [60996754800000, 61033647600000]) \\ &= \frac{\mid l_1 - l_2 \mid + \mid r_1 - r_2 \mid}{\mid l_2 - r_1 \mid} \\ &= 0,68 \end{aligned}$$

Distance Function for Times Similar like dealing with dates a distance function for time specifications can be defined. Let $t : D(\tau_7) \rightarrow \mathbb{R}^+$ be a mapping which assigns each time specification $p \in D(\tau_7)$ to a number $n \in \mathbb{R}^+$ under preservation of the order. Let t be a possible mapping which maps the time specification s to the number of seconds (s) and milliseconds (ms) the time specification contains.

The given time specifications $s_1, s_2 \in D(\tau_7)$ are to be compared. Now a distance function for time specifications is defined as

$$d_t(s_1, s_2) := d_n(t(s_1), t(s_2))$$

where d_t denotes the distance function for time specifications to be defined and d_n denotes the distance function for numbers defined in 4.5.3 respectively.

Distance Computations for ProperNames For the instances of type ProperName (τ_8) only exact matches are defined. Let $n_1, n_2 \in D(\tau_8)$ then the distance

$$d_{pn}(n_1, n_2) := \begin{cases} 0 & \text{if } n_1 = n_2 \\ \infty & \text{if } n_1 \neq n_2 \end{cases}$$

It is easy to see that this function fulfills the properties of a distance function.

Example 4.5.41

It is trivial that the distance of two ProperNames (strings) is set to 0 if the string compare function of the two ProperName strings results equality and to set the distance to `maxint` if the string compare function results inequality.

4.5.4 Aggregate Functions for Complex Types

Another issue to consider is the computation of combined distance measures. These are introduced to specify the distance between two instances of complex types, so called complex values. Let an RDS or ODS be given. Now the level of the RDS or ODS is computed recursively using the level function

$$l(RDS) = 1 + max_{i=1,\dots,n} l(\tau_i)$$

(or $l(ODS)$ respectively) from Section 4.4.1, where $\{\tau_i \mid i = 1, \dots, n\}$ are all types which occur in the definition of RDS and ODS.

Once the level $\chi_{RDS} := l(RDS)$ of RDS (ODS respectively) is computed the process of matchmaking starts. The distance measures between the atomic values in level 0 are computed. The resulting distance measures are inserted into level 1. This process goes on until level χ_{RDS} is reached.

Definition 4.5.42 ***Combined Distance Function, Combined Distance Measure***

Let $\tau \in \mathcal{T} \backslash \mathcal{B}$ be a complex type. Let $\tau_1, \dots, \tau_m \in \mathcal{T}$ be the types which occur in the definition of τ. Let $d_1, \dots, d_m$ be the distance functions for these types. Then $d : D(\tau) \times D(\tau) \to \mathbb{R}^+$ is called **combined distance function** for type τ if

(i) $d(v_1, v_2) = 0 \Longleftrightarrow v_1 = v_2$ for all $v_1, v_2 \in D(\tau)$
(ii) $d(v_1, v_2) = d(v_2, v_1)$ for all $v_1, v_2 \in D(\tau)$

hold. If additionally

(iii) $d(v_1, v_3) \leq d(v_1, v_2) + d(v_2, v_3)$ for all $v_1, v_2, v_3 \in D(\tau)$

holds, d is called **combined metric**.

Let $v_j, v_r \in D(\tau)$ be two complex values where v_j is taken from candidate $OF_j \in \mathcal{OF}$ and v_r is taken from centroid RQ. Then $\delta_j := d(v_r, v_j) \in \mathbb{R}^+$ is called **combined distance measure with respect to** RQ between candidate OF_j and centroid RQ.

The existence of a combined distance function for each complex type $\tau \in \mathcal{T} \backslash \mathcal{B}$ is not postulated. Instead for each type one default distance function is defined and alternative distance functions are given.

Combined distance functions can be defined very differently. For various reasons this definition can vary. Arguments to propose different distance functions for the same complex type are:

1. Higher or lower complexity for computations of distance measures (time needed for computation).
2. Weighting of different aspects.
3. Integration of k.o. criteria.
4. Integration of threshold criteria.

The choice of the right combined distance measure depends on the application properties. Some different ways to compute combined distance measures are introduced in Section 4.5.5.

In the following sections the default combined distance functions are introduced for each complex type. Every time a distance function is defined for a complex type τ, it is assumed that distance functions for every type $\tau_1, \ldots, \tau_m$ which occur in the definition of type τ are defined already.

For lists, arrays and records the default distance functions as well as the additionally defined measures are based on the arithmetic average. Different average calculations like harmonian or geometric average calculations are possible.

Distance Computation for Sets Sets are the most unstructured type considered in Grappa. For that reason it is most difficult to provide a powerful distance function. Consider the domain $D(\tau)$ of a complex type τ which is inductively defined by $\tau := \{\tau_i\}$. Without loss of generality a distance function is asumed which is defined on $D(\tau_i)$.

Having a distance function for each possible element of the considered sets e.g. the results which were obtained in Eiter and Mannila (1997) can be applied. Let v_1, v_2 be elements of $D(\tau)$.The link distance can be applied as a default combined distance function for the complex type set.

Definition 4.5.43 *Link Distance*

Let $\tau \in \mathcal{T} \backslash \mathcal{B}$ be a set type of type $\tau_i \in \mathcal{T} \backslash \{\tau\}$. Let $v_1, v_2 \in D(\tau)$ be two instances of τ. A linking between v_1 and v_2 is a relation $R \subseteq v_1 \times v_2$ which satisfies
(i) for all $u_1 \in v_1$ there exists $u_2 \in v_2$ such that $(u_1, u_2) \in R$.
(ii) for all $u_2 \in v_2$ there exists $u_1 \in v_1$ such that $(u_1, u_2) \in R$.
The minimum link distance d between subsets $v_1 \in D(\tau)$ and $v_2 \in D(\tau)$ is defined by

$$d(v_1, v_2) := \min_R \sum_{(u_1,u_2) \in R} d_i(u_1, u_2)$$

where d_i, $i = 1, \ldots, n$ are element distance functions and the minimum is taken over all relations R such that R is a linking between v_1 and v_2. Consequently, R are all possible allocations between the two sets v_1 and v_2 written as pairs of elements.

Distance Computation for Lists The list type is more structured than the set type. A type τ which is a list may be given. This type is defined as $\tau := \tau_i^+$. Let further be d_i a distance function for type τ_i.

The list type has the property that the length is not determined. So two values $v_1, v_2 \in D(\tau)$ can have different lengths $len(v_1) = \iota_1$ and $len(v_2) = \iota_2$. As a default combined distance function for lists the average of the pairwise comparison up to the length ι_i of the shorter list is defined.

Definition 4.5.44 *Distance Measure for Lists*

Let $\tau \in \mathcal{T} \backslash \mathcal{B}$, $\tau := \tau_i^+$ be a list type of element type $\tau_i \in \mathcal{T} \backslash \{\tau\}$. Let d_i be the distance function for type τ_i. Let $v_1, v_2 \in D(\tau)$ be two lists. Let

$$\iota_{\min} := \min_{i \in \{1,2\}} len(v_i)$$

be the length of the shortest list. Let $v_1 := \langle u_1^1, \ldots, u_1^n \rangle$ and $v_2 := \langle u_2^1, \ldots, u_2^m \rangle$ where $u_j^k \in D(\tau_i)$ for $j \in \{1,2\}$, $k \in \{1, \ldots, n\}$. Let $n > m$. Let $\varepsilon \in \mathbb{R}$ with $0 < \varepsilon \ll 1$. Then the distance between $v_1, v_2 \in D(\tau)$ is defined as

$$d_l(v_1, v_2) := \begin{cases} \frac{\sum_{i=1}^{\iota_{\min}} d(u_1^i, u_2^i)}{\iota_{\min}} & \text{if } len(v_1) = len(v_2) = l_[min] \\ \max\{\frac{\sum_{i=1}^{\iota_{\min}} d(u_1^i, u_2^i)}{\iota_{\min}}, \varepsilon\} & \text{otherwise.} \end{cases}$$

Here and in all following distance function definitions the ε is introduced to avoid 0-distances. 0-distances provide disadvantages in handling complex matchmaking applications due to the fact that they might not be caused by badly matching data but by incomplete instances to be matched.

It is obvious that d_l fulfills the properties of a distance function if the properties are fulfiled by the distance functions of the subdimensions (d). This combined distance measure provides a default distance measure for lists.

Distance Computation for Arrays For arrays the distance computation is fairly easy. The reason is that there are always the same number of entries of the same type. I case of a type $\tau \in \mathcal{T}$ which is an array of n entries of type $\tau_i \in \mathcal{T}$. Let d_i further be a distance function which computes the distance among values $u_1, u_2 \in D(\tau_i)$.

Now a default distance between two complex values $v_1, v_2 \in D(\tau)$ can be defined.

Definition 4.5.45 ***Distance Function for Arrays***

Let $\tau \in \mathcal{T}\backslash\mathcal{B}$, $\tau := \tau_i[1:n]$ be an array type of element type $\tau_i \in \mathcal{T}\backslash\{\tau\}$. Let d_i be a distance function for type τ_i. Let $v_1, v_2 \in D(\tau)$ be two arrays. Let $v_1 := [u_1^1, \ldots, u_1^n]$ and $v_2 := [u_2^1, \ldots, u_2^n]$ where $u_j^k \in D(\tau_i)$ for $j \in \{1, 2\}$, $k \in \{1, \ldots, n\}$. Then the distance between $v_1, v_2 \in D(\tau)$ is defined as

$$d_a(v_1, v_2) := \frac{\sum_{i=1}^{n} d(u_1^i, u_2^i)}{n}.$$

This default distance function simply computes the average of all distances between the elements of the arrays.

Distance Computation for Records The distance computation among records is as easy as the distance computation for arrays. The difference is that records may contain entries of different types. Let $\tau \in \mathcal{T}$ be a record. Let further assume that this record consists of types $\tau_1 \in \mathcal{T}, \ldots, \tau_n \in \mathcal{T}$. Under the condition that distance functions for the instances of types $\tau_1, \ldots, \tau_n$ are defined further distance functions for records can be defined as follows.

Definition 4.5.46 ***Distance Function for Records***

Let $\tau \in \mathcal{T}\backslash\mathcal{B}$, $\tau := (\tau^1, \ldots, \tau^k)$ be a record type of length k which consists of element types $\tau^i \in \mathcal{R}$ for $i = 1, \ldots, k$, where $\mathcal{R}$ is the set of types $\tau_i \in \mathcal{T}\backslash\{\tau\}$ $i = 1, \ldots, n$ ($n \leq k$) which occur in the definition of τ.
Let $d_1, \ldots, d_n$ be the distance functions for the types $\tau_1, \ldots, \tau_n$. Let $v_1, v_2 \in D(\tau)$ be two records. Let $v_1 := (u_1^1, \ldots, u_1^k)$ and $v_2 := (u_2^1, \ldots, u_2^k)$ where $u_j^l \in D(\tau^l)$, $l \in \{1, \ldots, k\}$, $j \in \{1, 2\}$. Then the distance between the records $v_1, v_2 \in D(\tau)$ is defined as

$$d_r(v_1, v_2) := \frac{\sum_{i=1}^{k} d_i(u_1^i, u_2^i)}{k}.$$

This provides a default combined distance function for records. Different possible combined distance functions for records are introduced in Section 4.5.5.

4.5.5 Weighting Paradigms for Complex Matchmaking

In this section different alternative combined distance functions for arrays and records are introduced. If the input distance functions have the distance function properties listed in Definition 4.4.18, these are preserved under the application of all combined distance functions. If the input distance functions have also the metric property this is preserved by the link distance as well as the average functions introduced above. The weighted average combined distance function, which is introduced in this section also preserves the metric properties. The distance function for lists which is introduced above as well as the minimum measure method and the threshold weighting method do not preserve the metric properties of the input distance functions.

Minimum Measure Weighting Method In the minimum measure method only the closest entry of an array or record is regarded. A so called pre-matchmaking can be launched where all candidates are disregarded which do not even have a single entry which is close enough to the centroid.

Definition 4.5.47 ***Minimum Measure Method***

1. For **arrays:**
   Let $\tau \in \mathcal{T}\backslash\mathcal{B}$, $\tau := \tau_i[1:n]$ be an array type of element type $\tau_i \in \mathcal{T}\backslash\{\tau\}$. Let d_i be a distance function for type τ_i. Let $v_1, v_2 \in D(\tau)$ be two arrays. Let $v_1 := [u_1^1, \ldots, u_1^n]$ and $v_2 := [u_2^1, \ldots, u_2^n]$ where $u_j^k \in D(\tau_i)$ for $j \in \{1,2\}$, $k \in \{1,\ldots,n\}$. Let $\varepsilon \in \mathbb{R}$ with $0 < \varepsilon \ll 1$. Then the **minimum measure distance** between $v_1, v_2 \in D(\tau)$ is defined as

$$d_{a_m}(v_1,v_2) := \begin{cases} 0 & \text{if } v_1 = v_2. \\ \max\{\min_{i=1,\ldots,n} d(u_1^i, u_2^i), \varepsilon\} & \text{otherwise.} \end{cases}$$

2. For **records:**
   Let $\tau \in \mathcal{T}\backslash\mathcal{B}$, $\tau := (\tau^1, \ldots, \tau^k)$ be a record type of length k which consists of element types $\tau^i \in \mathcal{R}$ for $i = 1, \ldots, k$, where $\mathcal{R}$ is the set of types $\tau_i \in \mathcal{T}\backslash\{\tau\}$ $i = 1, \ldots, n$ $(n \leq k)$ which occur in the definition of τ.
   Let $d_1, \ldots, d_n$ be the distance functions for the types $\tau_1, \ldots, \tau_n$. Let $v_1, v_2 \in D(\tau)$ be two records. Let $v_1 := (u_1^1, \ldots, u_1^k)$ and $v_2 := (u_2^1, \ldots, u_2^k)$ where $u_j^l \in D(\tau^l)$, $l \in \{1,\ldots,k\}$, $j \in \{1,2\}$. Let $\varepsilon \in \mathbb{R}$ with $0 < \varepsilon \ll 1$. Then the **minimum measure distance** between $v_1, v_2 \in D(\tau)$ is defined as

$$d_{r_m}(v_1,v_2) := \begin{cases} 0 & \text{if } v_1 = v_2. \\ \max\{\min_{i=1,\ldots,k} d(u_1^i, u_2^i), \varepsilon\} & \text{otherwise.} \end{cases}$$

Weighted Average Weighting Method The weighted average method is based on the idea that some entry of an array or record has a higher relevance than others. For this reason it is possible to introduce weights for every entry of an array or record.

Definition 4.5.48 ***Weighted Average Method***

1. For **arrays:**
   Let $\tau \in \mathcal{T}\backslash\mathcal{B}$, $\tau := \tau_i[1:n]$ be an array type of element type $\tau_i \in \mathcal{T}\backslash\{\tau\}$. Let d_i be a distance function for type τ_i. Let $v_1, v_2 \in$

$D(\tau)$ be two arrays. Let $v_1 := [u_1^1, \dots, u_1^n]$ and $v_2 := [u_2^1, \dots, u_2^n]$ where $u_j^k \in D(\tau_i)$ for $j \in \{1,2\}$, $k \in \{1, \dots, n\}$. Let $w_i \in \mathbb{R}^+ \backslash \{0\}$, $i = 1, \dots, n$ be weights for the entries of τ. Then the **weighted average distance** between $v_1, v_2 \in D(\tau)$ is defined as

$$d_{a_w}(v_1, v_2) := \frac{\sum_{i=1}^n w_i d(u_1^i, u_2^i)}{\sum_{i=1}^n w_i}.$$

The default for w_i is 1.

2. For **records:**
Let $\tau \in \mathcal{T} \backslash \mathcal{B}$, $\tau := (\tau^1, \dots, \tau^k)$ be a record type of length k which consists of element types $\tau^i \in \mathcal{R}$ for $i = 1, \dots, k$, where $\mathcal{R}$ is the set of types $\tau_i \in \mathcal{T} \backslash \{\tau\}$ $i = 1, \dots, n$ $(n \leq k)$ which occur in the definition of τ.
Let $d_1, \dots, d_n$ be the distance functions for the types $\tau_1, \dots, \tau_n$. Let $v_1, v_2 \in D(\tau)$ be two records. Let $v_1 := (u_1^1, \dots, u_1^k)$ and $v_2 := (u_2^1, \dots, u_2^k)$ where $u_j^l \in D(\tau^l)$, $l \in \{1, \dots, k\}$, $j \in \{1,2\}$. Let $w_i \in \mathbb{R}^+ \backslash \{0\}$, $i = 1, \dots, k$ be weights for the entries of τ. Then the **weighted average distance** between $v_1, v_2 \in D(\tau)$ is defined as

$$d_{r_w}(v_1, v_2) := \frac{\sum_{i=1}^k w_i d(u_1^i, u_2^i)}{\sum_{i=1}^k w_i}.$$

The default for w_i is 1.

With these weighted average distance functions it is possible to give certain entries of arrays and records higher relevance.

Threshold Weighting Method Let $\tau \in \mathcal{T}$ be an array or record. Let $v_r, v_j \in D(\tau)$ be complex values of type τ. The idea at the threshold method is to regard the distance measure of an entry $u_i \in D(\tau_i)$ of the array or record v_j to the array or record v_r only if it does not exceed a certain threshold.

Definition 4.5.49 ***Threshold method***

1. For **arrays:**
Let $\tau \in \mathcal{T} \backslash \mathcal{B}$, $\tau := \tau_i[1:n]$ be an array type of element type $\tau_i \in \mathcal{T} \backslash \{\tau\}$. Let d_i be a distance function for type τ_i. Let $v_1, v_2 \in D(\tau)$ be two arrays. Let $v_1 := [u_1^1, \dots, u_1^n]$ and $v_2 := [u_2^1, \dots, u_2^n]$ where $u_j^k \in D(\tau_i)$ for $j \in \{1,2\}$, $k \in \{1, \dots, n\}$. Let $\delta_1, \dots, \delta_m$, $m \leq n$ be the distance measures $d(u_1^i, u_2^i)$ $i = 1, \dots, n$ which are less or equal to a threshold value $th \geq 0$. Let $\varepsilon \in \mathbb{R}$ with $0 < \varepsilon \ll 1$. Then the **threshold distance** between $v_1, v_2 \in D(\tau)$ is defined as

$$d_{a_t}(v_1, v_2) := \begin{cases} 0 & \text{if } v_1 = v_2 \\ \max\{\frac{\sum_{i=1}^m \delta_i}{m}, \varepsilon\} & \text{otherwise.} \end{cases}$$

2. For **records:**
Let $\tau \in \mathcal{T}\backslash\mathcal{B}$, $\tau := (\tau^1, \ldots, \tau^k)$ be a record type of length k which consists of element types $\tau^i \in \mathcal{R}$ for $i = 1, \ldots, k$, where $\mathcal{R}$ is the set of types $\tau_i \in \mathcal{T}\backslash\{\tau\}$ $i = 1, \ldots, n$ $(n \leq k)$ which occur in the definition of τ.
Let $d_1, \ldots, d_n$ be the distance functions for the types $\tau_1, \ldots, \tau_n$. Let $v_1, v_2 \in D(\tau)$ be two records. Let $v_1 := (u_1^1, \ldots, u_1^k)$ and $v_2 := (u_2^1, \ldots, u_2^k)$ where $u_j^l \in D(\tau^l)$, $l \in \{1, \ldots, k\}$, $j \in \{1, 2\}$. Let $\delta_1, \ldots, \delta_m$, $m \leq n$ be the distance measures $d_i(u_1^i, u_2^i)$ $i = 1, \ldots, k$ which are less or equal to a threshold value $th \geq 0$. Let $\varepsilon \in \mathbb{R}$ with $0 < \varepsilon \ll 1$. Then the **threshold distance** $v_1, v_2 \in D(\tau)$ is defined as

$$d_{r_t}(v_1, v_2) := \begin{cases} 0 & \text{if } v_1 = v_2 \\ \max\{\frac{\sum_{i=1}^m \delta_i}{m}, \varepsilon\} & \text{otherwise.} \end{cases}$$

The threshold distance method presents the possibility of considering only the values (inside a specific candidate or centroid instance) which are close enough to the provided candidate or centroid instance. It is obvious that the threshold distance function fulfills the properties of a distance function.

4.6 Summary

This chapter provides an open, expandable but at the same time sound mathematical framework for multidimensional matchmaking in electronic negotiation scenarios. The central notion of the Grappa matchmaking system (GMS) based on the structured Grappa matchmaking base (SGMB) are introduced. These concepts rely on the following – most important – components.

- Distance function and metric definitions for basic and complex types.
- The mapping MAP between heterogeneous offer and request structures.
- An agent based role model, enabling actor-centered knowledge and decision policies.

Currently the matchmaking process is limited to $1 : n$ (centroid:candidate) matchmaking operations. Although artificial $n : m$ operations can be simulated by executing m $1 : n$ operations successivley this operation can be enhanced. The matchmaking algorithms have the potential to provide a $n : m$ matching. Due to the high computational complexity of those operations pruning algorithms to limit the search space are recommended here (see also Subrahmanian et al. 2000).

The independent formalization of the matchmaking problem in this chapter provides a basis for the implementation in programming languages.

In the consecutive Chapter 5 the implementation of the Grappa prototype whithin the dynamic, expandable matchmaking library `de.siemens.zt.ik6-matching` is introduced. This prototype is based on state-of-the-art programming paradigms such as JAVA, EJB and RMI.

5 Matchmaking Implementation

The implementation of matchmaking is the step towards applying the matchmaking algorithms defined so far in practice. The aim of this chapter is to show the ideas behind the implementation of the generic matchmaking framework rather then providing a detailed code listing.

In Section 5.1 the architectural ideas are described in general. In Section 5.2 the applied technologies are shortly introduced. Here JAVA, RMI-Server and EJB technology is briefly introduced which may be skipped by the advanced reader. In Section 5.3 the implementation based on the provided technologies is concerned. Here excerpts from application example as well as headers and interfaces of the central GRAPPA methods are provided. Finally Section 5.4 finishes the chapter.

5.1 Scope

The generic matchmaking implementation has different features and desiderata which must be fulfilled to apply it to various electronic negotiation scenarios.

- Adaptability: The architecture must allow a system designer to adapt matchmaking processes according to offer and request structures.
- Heterogeneity: The architecture must allow to model heterogeneous offer and request profiles as modelled in the last chapter.
- Portability: The architecture should be able to run on heterogeneous systems. This includes that the matchmaker can operate on data which are not supported from the same system. Also the matchmaker should be transferrable to another system.
- Scalability: The architecture must be scalable to different application scenario sizes and performance demands.
- Distribution: The architecture must allow to integrate candidates and centroids from heterogeneous sources, i.e., software agents, Multi-Agent Systems and databases to support candidates and centroids.

Adaptability and heterogeneity can be handled by applying XML as configuration language. XML, the eXtensible Markup Language which has succeeded as standard definition language for electronic data exchange formats on the web supports Document Type Definitions (DTD's). Each concrete DTD provides the structure for a class of objects. In case of the generic matchmaking library XML-DTD's are applied to define the general structure of all configurations (i.e. mappings, see MAP in Section 4.4.3) as well as the structure of the candidates and centroids. In case of the candidates and centroids a concrete DTD is provided by a system designer for each class of candidates or centroids.

The portability is a central point in todays software architectures. To provide a portable library Grappa has been implemented using the JAVA programming language. JAVA compilers and resources are today available on nearly every common operating system. The central tradeoff using JAVA for a large scale software project is the lack of performance in comparison to other object oriented programming languages like C++ or Small Talk. This problem is overcome by the possibility of scalable applications in matchmaking. The Enterprise Java Beans (EJB) technology enables load balancing mechanisms in large scale applications which helps to flatten peaks and reduce the load on single processes within the application.

Distribution plays a central role in todays market based applications. For this reason the MECCA Agent platform is applied in the Grappa application to encapsulate the offer and request profiles. These profiles which are represented by the centroid and the candidate agent are matched against each other within the Grappa matchmaker. The MECCA agents do play several roles. They can either encapsulate one profile per agent or wrap an entire database. Each agent can in this way provide either a single offer or request or represent a set of candidates or centroids within the system.

In the same context the Remote Method Invocation (RMI) server is applied. The RMI server is no integral component of the matchmaking library. The matchmaking library is able to run completely stand-alone as a pure JAVA application. Alternatively the matchmaking library components can be deployed as beans on the EJB server. In such a scenario the matchmaker can be queried very flexibly. In the prototype implementation of the application the RMI server technology is introduced to provide a central unit which processes the centroids and times on the one hand the matchmaker load and on the other the candidate and centroid schedules.

5.2 Applied Technology

In the next section the technologies (i) JAVA, (ii) RMI server and (iii) EJB are briefly introduced. The central idea is to provide an overview perspective for readers which are interested in implementation issues but are not familiar with JAVA technologies.

5.2.1 JAVA

In Kalin (2001) the authors introduce the JAVA programming language. Since the offspring of JAVA which has been developed by Sun Microsystems and released in a first version in April 1991 as the successor of C++ numerous books and introductions about this object-oriented, easy-to-use and portable programming language have been written. Java is a general-purpose programming language which provides a rich assortment of data types, operators, control structures, and standard packages. Java supports all state-of-the-art programming paradigms such as arrays, lists, vectors, records, operators such as loops in while, for and do-while statements. As a more advanced example JAVA provides graphic libraries such as the Abstract Window Toolkit (AWT) and the Swing set. The Swing set is used to implement the HrNetAgent prototype GUI agents. As a central aspect the JAVA programming language is object-oriented and provides all key features of such a language.

JAVA provides primitive data types such as `int` and `double` which do not fully support the object-oriented paradigm but accelerate the computation time. Each program in JAVA requires at least one class. All functions are encapsulated in classes as either constructors or methods. A constructor is a method to "instantiate" a class into an object. Variables are either encapsulated as fields or local variables in constructors and methods. JAVA has only single inheritance for classes but a class may implement arbitrarily many interfaces. Multiple inheritance is omitted due to the complex inheritage structure resulting. In the case of interfaces multiple inheritance is enabled which substitutes the multiple inheritance on class level and makes programming less complex. Methods and fields can be associated either with a class itself or with class instances or objects. Polymorphism and other object-oriented constructs are applied within JAVA's standard classes widely.

An improvement of program development in JAVA can be achieved by applying Uniformed Modelling Language (UML) for object oriented modelling and derive JAVA code from the UML diagrams.

Standard packages supported by the JAVA Development Kit (JDK) are packages supporting

- Graphics
- Networking
- Security
- Persistence
- Database access
- Reflection
- Components
- and many more

Furthermore, the javax packages are added as java extensions. Many excellent commercial and non-commercial packages can be obtained for the JAVA development environment. Implementing Grappa the JDK version 1.3.1 has been used.

5.2.2 RMI Server – Remote Method Invocation

An Remote Method Invocation (RMI) server is a component of the JAVA platform and is designed to execute methods on other machines or commands from other machines on the local machine. It runs as a separate process on the local computer and realizes an asynchronous message handling. An RMI server is not designed to run as a full fledged application server but it can be applied to handle a set of distributed software agents – such as the MECCA agents – which are coordinated within an application. In Harned (2001) the author is focusing on the issue of enabling RMI applications in asynchronous process managing applications. This article is also recommended for deeper discussion of the RMI server terminology.

In the Grappa framework the RMI server technology is used as a coordination tool among the agents. The RMI server keeps in contact with the matchmaker agent, the centroid and candidate agents and triggers matchmaking processes and communicates the requests, offers and results to the participants. In the Grappa implementation the RMI version 1.3.1 has been applied.

5.2.3 EJB – Enterprise Java Beans

Enterprise JAVA Beans are applications which are running server sided. An EJB is a piece of code that can be used by many different applications. It is located on an EJB server. The process of placing such a piece of generic code on a server is called "deploying". The EJB technology has been developed by Sun Microsystems based on the JAVA programming language. Sun also released the reference implementation of the first EJB server called Java 2 Enterprise Edition (J2EE). Since the release of J2EE many commercial and open-source projects started to develop EJB servers. The most important are the following.

- IBM's Websphere
- BEA's Weblogic
- Open source community's JBOSS

Although JAVA code, which has been provided for a specific EJB server, is generally portable to any other EJB server the standard routines and information which have to be provided for the individual server differ only slightly. In the following the information provided here is related to the JBOSS EJB server which has been used in version 2.2.1 for the Grappa implementation.

The process of deploying a provided piece of JAVA code as an EJB is performed as follows. Each application has to implement several java interfaces to be "deployable". The names of the means as well as the interface classes are specified in XML files which describe the application to the application server. Finally the compiled code-base as well as the XML (meta-) files are combined into a JAR file. A JAR file is a compressed library file containing the application code. This JAR file can finally be deployed on a JBOSS EJB server which provides the application to the system.

For details concerning EJB programming the reader is forwarded to Caple and Altarace (2001). There also exist numerous books on this topic, due to the fact that EJB and the corresponding Microsoft technology ASP.NET are the state-of-the-art middleware paradigms for web services and distributed applications.

5.3 Implementation of GRAPPA

In this section, the key issues of implementation of the matchmaking framework Grappa are described. The software framework `de.siemens.zt.ik6-matching` which is introduced here implements the Grappa architecture. In particular, the framework has been designed to be extensible and to be integrated easily into commercial e-business platforms.

5.3.1 Basic Entities and Processes

In this section, the basic computational concepts used in the implementation are described. Request instances are referred to as centroid profiles and offer instances as candidate profiles for the implementation issue. Consequently the centroid profile encapsulates the structure of the originating request. This structure is defined by means of an XML document type definition (DTD). centroids on the domain-specific matchmaker must conform to this DTD. Candidate profiles are the data instances on which the matchmaking will be processed (e.g. records in an applicants database). Alike the centroid profile, this structure must be defined in an XML document type definition. A simplified example of the candidate and centroid profile can be seen in Figure 5.1.

To compute a match between centroid and candidate profile based on various distance functions, a 1:1-mapping between the elements of the centroid and the candidate is required. Especially if the centroid and the candidate provide a different number of top-level dimensions or attributes a orthogonalization process is required which will be described in the following. In this process, attributes that together describe a separate semantic aspect relevant for matchmaking are determined and grouped into a cluster. An attribute may be contained in more than one cluster. If this is this case, they are duplicated to preserve orthogonality. The clustered scheme which is the result of the orthogonalization process is called "pseudoorthogonal".

Centroid roleprofile	**Candidate applicantprofile**
age	age
salary	salary
generalrequirements specialrequirements	generalskills specialskills

Figure 5.1. Simplified Example Candidate and Centroid Profile.

The matchmaking library provides a number of built-in distance functions that can be used to create domain-specific matchmakers. All distance functions will need to implement an interface DistanceFunction. This way, domain-specific distance functions implementing custom semantics required for specific matchmaking solutions can be provided.

Aggregate functions are used by the matchmaker to compute a multi-attribute result from the values returned by the basic distance functions. The matchmaking library contains the aggregate function weighted sum to compute a weighted sum of the basic distances as an overall result. Different aggregate functions implementing thresholds, hard constraints, or averages can be included similarly as additional distance functions using the interface AggregateFunction.

Figure 5.2 shows the example from Figure 5.1 after the orthogonalization and the association of the distance functions "number_distance", "free-text_distance" and "keyword_distance".

It is also distinguished between generic and domain-specific distance functions. The functions described so far are generic and can be used as building blocks for a wide range of matchmaking tasks. Some generic distance functions have also been introduced in Section 4.5. However, in order to perform good-quality matchmaking, it is necessary to add domain specific know-how. For example, in the human resources application described in Section 6, we use distance functions to determine the distance of a user's education, skills, and experience specification against corresponding elements of a job profile.

5.3.2 Configuration

In order to customize the generic matchmaker for a domain, a configuration process is provided: First, the structure of centroid (request to the match-

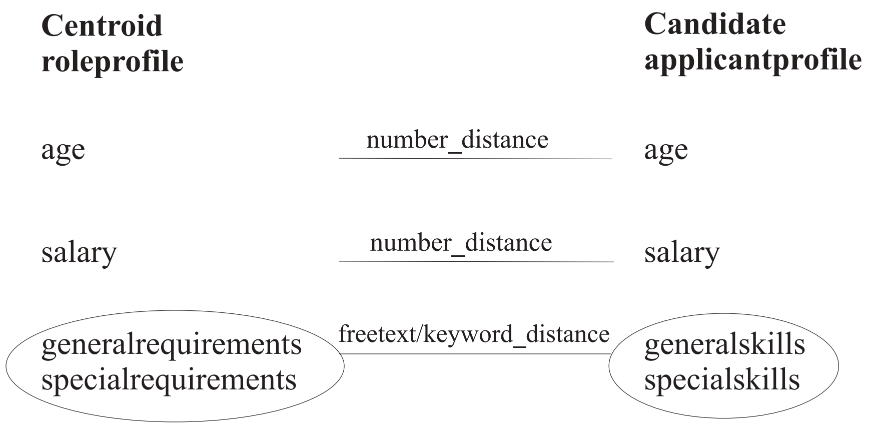

Centroid roleprofile
Candidate applicantprofile
age
number_distance
age
salary
number_distance
salary
generalrequirements specialrequirements
freetext/keyword_distance
generalskills specialskills

Figure 5.2. Simplified Example Candidate and Centroid Mapped and Orthogonalized.

maker) and candidate (data instances) profiles has to be defined. These structures must be provided as XML document type definitions (DTD). During the matchmaking process, instances of the centroid and candidate profiles must comply with these DTDs. Then, the main configuration file (XML document) has to be created. In this document, the clustering of attributes, and association of distance functions with each pair of centroid / candidate cluster pair are specified. Additionally this document contains the specification of the aggregate parameters whose meaning depends on the corresponding aggregate function (in the case of the weighted sum, the aggregate parameter would represent simply the weight).

Furthermore for each cluster pair, a constraint type (soft or hard) must be declared. This type determines the impact of an absolute inequality of a cluster-pair on the overall result. So a distance of 0 of one cluster compareison with the comparison type "strong" would lead to an overall result of 0 independent of any other comparison result. Each cluster and its associated distance function is considered as one dimension in the configuration. The configuration entries for each dimension are grouped as 5-tuples consisting of: (i) cluster from centroid; (ii) cluster from candidate; (iii) distance function; (iv) aggregate parameter; and (v) constraint type. As mentioned above, the computation of a distance value between two clusters can also be done by dividing each cluster into its subitems, compute the distance of these subitems directly and use an aggregate function to obtain an overall result for this cluster.

In this case the corresponding dimension entry in the configuration file would have a number of sub-dimension entries and instead of a distance function an aggregate function for the overall result has to be specified. The configuration file with a structure as described above is written in XML using a special configuration document type definition Configuration.dtd provided as a part of the matchmaking library. The dimension entries in configuration file for the example are shown in Table 5.1.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE Configuration SYSTEM "Configuration.dtd">
<Configuration>
    <LeftName>roleprofile</LeftName>
    <RightName>applicantprofile</RightName>
    <Dimension>
        <LeftName>professions</LeftName>
        <RightName>desiredjob</RightName>
        <RightName>jobhistory</RightName>
        <RightName>professions</RightName>
        <DistanceFunction>de.siemens.zt.ik6matching.
        distance.JobDistance</DistanceFunction>
        <Type>weak</Type>
        <AggregateParameter>4</AggregateParameter>
    </Dimension>
    <Dimension>
        <LeftName>experiencelevel</LeftName>
        <LeftName>professions</LeftName>
        <RightName>jobhistory</RightName>
        <DistanceFunction>de.siemens.zt.ik6matching.
        distance.JobHistoryDistance</DistanceFunction>
        <Type>weak</Type>
        <AggregateParameter>3</AggregateParameter>
    </Dimension>
    ...
    <AggregateFunction>de.siemens.zt.ik6matching.
    aggregate.WeightedSum</AggregateFunction>
    <ScoreParameter>0.7</ScoreParameter>
    <classes>
        <url>ik6matching.jar</url>
    </classes>
</Configuration>
```

Table 5.1. Example Configuration Instance for the Grappa Matchmaker

Using this configuration information, a domain-specific instance of the matchmaker can be created automatically.

5.3.3 Matchmaking JAVA-library `de.siemens.zt.ik6matching`

The matchmaking library consists of a number of classes covering configuration aspects such as dimension entries and the main configuration file, a class for the generic matchmaker which operates with the configuration and carries out the matchmaking process between instances of corresponding classes for centroid and candidates. The matchmaking result is covered by a class which provides detailed information about the matchmaking process such as information about each cluster-cluster comparison, reasons for "disqualifying" a candidate, output to XML, etc. The following paragraphs provide a short description of the most important classes included in the matchmaking library: The dimension-class encapsulates the dimension entry (a 5-tuple) in the configuration file with declaration of the clusters in centroid and candidate, distance function (resp. sub-dimensions an aggregate function), the aggregate parameter and type.

In Table 5.2 the individual components of the matchmaking library `de.siemens.zt.ik6matching` are shown. The matchmaker component within the library implements the JAVA interface shown in Table 5.3. This interface is one selection from three different interfaces for the core matchmaking procedure. The other interfaces are created to dynamically match centroid instances against all candidates which are supported by datasources which are connected to the system via candidate factories. These factory interfaces are not considered in the following because they refer to a special implementation paradigm which is very useful in EJB implementations but not relevant to the functionality of the matchmaking procedures.

`aggregate`	Aggregate functions for distance computation.
`common`	Interfaces.
`core`	Matchmaking application integration classes.
`distance`	Distance functions for atomic types.
`EJBmm`	The EJB matchmaker.
`EJBServices`	Additional EJB services, e.g. standard wrapper procedures.
`tools`	Matchmaking tools.
`util`	Factory classes and provider classes for data integration.

Table 5.2. `de.siemens.zt.ik6matching` Components

The config-class is the main configuration class, encapsulating the configuration file and administering the dimension entries. During construction, an object of this class loads all required distance and aggregate functions, creates dimension entries and builds the necessary internal structure for the

```
public interface Matching {
    public Match[ ] match(Centroid centroid, Candidate[ ]
candidate);
}
```

Table 5.3. JAVA-Interface "Match"

```
public interface Configure {
    public boolean configure(CentroidDTD cendtd,
CandidateDTD candtd, MMConfig mmconfig);
}
```

Table 5.4. JAVA-Interface "Configure"

matchmaker. Additionally this class provides the application with two factory objects for loading centroid and candidate data. Instances of the matching class must be initialized with a configuration object and will then perform the domain-specific matchmaking on centroid and candidates. Additionally this class can have a number of candidate providers from which candidates can be drawn.

The configuration class within the matchmaking application implements the configuration interface which is shown in Table 5.4.

The candidate provider interface automates the access of candidate data for the matchmaker. Implementing this class the user has the ability to preview and preselect candidates from various data sources before matchmaking so that unnecessary comparisons can be avoided.

In Figure 5.3 a class diagram for several classes and interfaces of the `de.siemens.zt.ik6matching` package is shown. The `ProfileFactory` defines the abstract superclass of the `CandidateFactory` and the `CentroidFactory`.

Each of those factories internally represents the source for the candidate and centroid instances which are currently available. The `CandidateProvider` class defines a source for new candidates. This source can either be a file, a network resource (e.g. database) or agent.
Finally the superinterface `Function` which provides the interfaces `AggregateFunction` and `DistanceFunction` which have to be implemented by any *atomic* and *complex distance function* which is integrated into the `de.siemens.zt.ik6matching` package.

In Figure 5.4 the structure of the implementation of the GRAPPA system is shown. In this simplified structural version of the system the processes which have been introduced in theory in Section 2.4 are applied in practice.

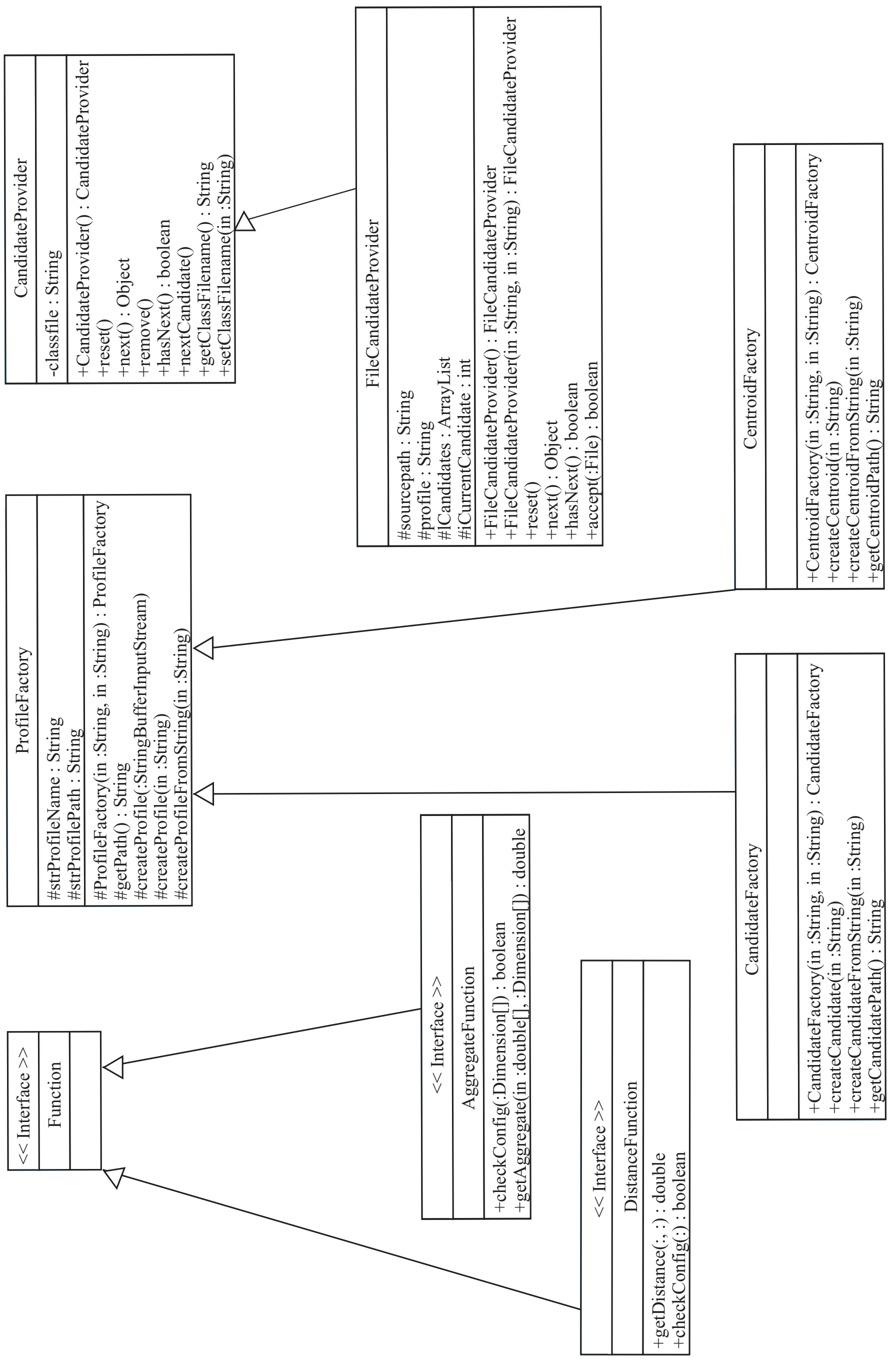

CandidateProvider
-classfile : String
+CandidateProvider() : CandidateProvider
+reset()
+next() : Object
+remove()
+hasNext() : boolean
+nextCandidate()
+getClassFilename() : String
+setClassFilename(in :String)
FileCandidateProvider
#sourcepath : String
#profile : String
#lCandidates : ArrayList
#iCurrentCandidate : int
+FileCandidateProvider() : FileCandidateProvider
+FileCandidateProvider(in :String, in :String) : FileCandidateProvider
+reset()
+next() : Object
+hasNext() : boolean
+accept(:File) : boolean
ProfileFactory
#strProfileName : String
#strProfilePath : String
#ProfileFactory(in :String, in :String) : ProfileFactory
#getPath() : String
#createProfile(:StringBufferInputStream)
#createProfile(in :String)
#createProfileFromString(in :String)
CentroidFactory
+CentroidFactory(in :String, in :String) : CentroidFactory
+createCentroid(in :String)
+createCentroidFromString(in :String)
+getCentroidPath() : String
CandidateFactory
+CandidateFactory(in :String, in :String) : CandidateFactory
+createCandidate(in :String)
+createCandidateFromString(in :String)
+getCandidatePath() : String
<< Interface >>
Function
<< Interface >>
AggregateFunction
+checkConfig(:Dimension[]) : boolean
+getAggregate(in :double[], :Dimension[]) : double
<< Interface >>
DistanceFunction
+getDistance(:, :) : double
+checkConfig(:) : boolean

Figure 5.3. Class-Diagram for an Excerpt of Classes in `de.siemens.zt.ik6matching`.

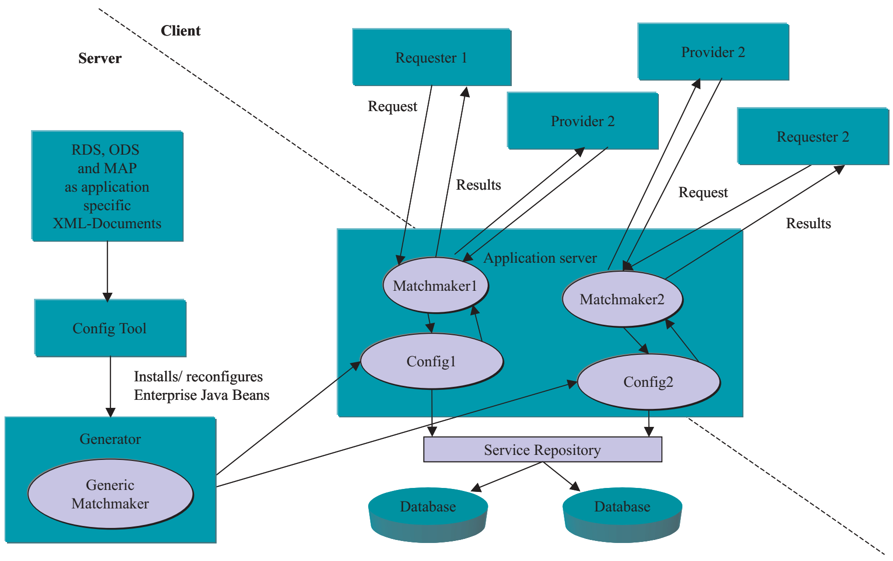


Figure 5.4. GRAPPA Implementation Overview.

1. In the first step, the application dependent RDS, ODS and MAP are provided as application dependent XML-Documents for the matchmaker using a Config Tool.
2. The Config Tool installs and reconfigures the matchmaker as Enterprise Java Beans on the matchmaking generator, instantiating a Generic Matchmaker.
3. The Generic Matchmaker installs the application dependent configuration from (1.) on the application Server (Matchmaker1, Config1, Matchmaker2, Config2).
4. Runtime, requesters (Requester 1, Requester 2) and providers (Provider 1, Provder 2) which are participating software agents (centroids and candidates) can access the matchmaking service.

5.3.4 Deployment as Enterprise Java Beans Component

In addition to the classes of the basic matchmaker described above the library provides a set of classes for the deployment of a domain-specific matchmaker in an application server to provide matchmaking functionality in the context of a commercial e-business platform and to make it available from the web or from other clients. These classes are two wrapper classes for the MMConfig and the matching class with some extra functionality suitable for EJB components. They are only briefly characterized here. Concrete implementation details of the individual classes and wrapper classes can be obtained from the javadoc comments of the `de.siemens.zt.ik6matching` package.

`Config`, `ConfigEJB` and `ConfigHome` are classes which implement wrappers for the matchmaker config class ConfigBean. This enterprise java bean is realized as an entity bean to provide a persistent storage for domain-specific configurations of the matchmaker.

`Matchmaker`, `MatchmakerEJB` and `MatchmakerHome` are classes which represent the MatchmakerBean session bean supporting the core matchmaking functionality. In combination with an instance of the ConfigBean this class performs the matchmaking in the application server.

5.4 Summary

This chapter finishes Part II by giving an overview on the implementation of the GRAPPA matchmaking framework which has been presented in Chapter 4. Because the aim of this chapter is not to provide full implementation details the interested reader is referred to the javadoc implementation reference for further information.

In this chapter the scope of the software architecture is described. Furthermore the applied technology is introduced which provides background on current implementation tools. Finally the concrete implementation of GRAPPA is introduced in Section 5.3. Here, four parts are distinguished. Firstly basic entities and processes in matchmaking implementations are regarded. Special attention is spent to the configuration issues of matchmaking applications. Finally the matchmaking library is described. The EJB component deployment complements the section.

In Part III the application and evaluation of the matchmaking framework is discussed. In Chapter 6 the application of GRAPPA in a real-world market – the human resources market – is introduced. Chapter 7 provides the methodology and the results of the GRAPPA evaluation in the German HR market. The evaluation results show the superiority of GRAPPA towards state-of-the-art matchmaking solutions.

6 Application in a Real-World Market

In this chapter the application of Grappa in a real-world market is described. The multidimensional matchmaking framework has been applied in an industrial project in the human resources market in Germany. The project called HrNetAgent is a prototype implementation of multidimensional matchmaking between candidates and centroids represented by employees and open positions in the german human resources market.

This chapter divides into the following sections. Section 6.1 comprises the project background. HR information systems as well as Internet based job exchanges are shortly described. Here, the individual specifics of the human resources market in Germany are regarded. In Section 6.2 the project background is briefly discussed. In Section 6.3 the HrNetAgent system is focussed. Here the system overview as well as the adoptions of Grappa which are application dependent are considered. Finally the chapter is summarized in Section 6.4.

6.1 Scope

In recent years a growing discipline between the fields of human resources and computer science has established. Traditionally HR experts focussed on psychologic questions of managing personnel. In computer science mainly the implementation and organization of accounting of human resources was in focus. During the past few years the electronization of many domains of society also caught the personnel management domain.

In this section two central tendencies towards the integration of human resources and computer science are presented. As the first one, human resources information systems (HRIS) are introduced in Section 6.1.1. As the second one, job exchanges in combination with corporate networks are provided in Section 6.1.2.

6.1.1 Human Resource Information Systems

As to Mülder (2000), Human Resources Information Systems (HRIS) are mainly applied in large enterprises (with more then 100 employees). In most application areas they are integrated into enterprise resource management systems (ERP-systems). In 1999 only 49 percent of the large enterprises in Germany applied a HR module with features beyond accounting within their ERP-system. The systems applied in these enterprises are HRIS systems providing the following key features beyond accounting: Time-management, personnel recruiting and development.

Generally said today's HR solutions in ERP-systems mainly provide administrative support and do not integrate intelligent recruiting or personnel clustering functionalities. In certain insulated areas, AI methods have been implemented to classify personnel and tasks. But these solutions have not yet been integrated into ERP-systems due to the fact that they are special cases and have not been generalized for domain independent applications.

Especially in the recruiting, market negotiation mechanisms are to be integrated into ERP-systems. As in many domains which are focusing electronization of traditionally manually solved problems, there are numerous obstacles to overcome. Certainly a central obstacle is the existence of many different laws which enable only heterogeneous HRIS in different countries. Another crucial point is the acceptance of e-business solutions to recruiting procedures by both, the HR departments as well as the applicants in the market. Finally an important issue is the existence of non-integrated accounting solutions which are present in many companies.

It is useful to integrate human resources applications into ERP systems. One reason is the simplification of the allocation of internal advertisements of vacancies. Often, especially in large enterprises, human resources are hard to control because standardized applications are missing. To integrate powerful human resources mechanisms into ERP systems matchmaking can be used. Additionally to the above mentioned advantages, the integration of human resources planning into ERP systems also makes project costing easier.

Today, standards like HR-XML[1] and application defined XML dialects enable the system independent and general integration of electronic employment marketplaces. Methods like intelligent matchmaking enable search and ranking possibilities far beyond simple keyword search algorithms such as integrated into todays personnel planning tools.

6.1.2 Job Exchanges and Corporate Networks

More and more large enterprises use electronic job exchanges for the recruitment of personnel. Here both, integrated solutions within the corporate network of a company as well as external personnel services such as job exchanges on the Internet like Monster[2] or Stepstone[3] offer certain integration

[1] http://www.hr-xml.org (07/16/2002)
[2] http://www.monster.com (06/25/2002)
[3] http://www.stepstone.com (06/25/2002)

features for company networks. Here again the HR-XML standard enables standardized communication and opens flexible perspectives for future recruiting processes.

Today many large companies integrate directly online application forms into their web pages. These application forms have several advantages and disadvantages. From the perspective of the company the process of application is strongly simplified if the applicant's profiles are available in an electronic format. The recruiting process is fastened and the application administration process simplified. A central problem of today's recruiting processes is that both, electronic and traditional recruiting has to be performed. The resulting heterogeneous formats of the applications – some are electronic and other are submitted in traditional print format – disables a homogenous recruiting procedure and slows down the entire process. In future a pure electronic recruitment process is planned by many companies. This will of course enable new perspectives (see also Mülder 2000).

From the applicants point of view electronic recruiting also has different advantages and disadvantages. Certainly positive is the possibility for applications to submit the applications any time and they are cheaper (printing costs, photo duplication and mail costs are saved). Also the process is mostly more transparent and takes place in a higher speed. In non-integrated systems a strong disadvantage is that the applicant has to submit his/her details to many different systems. The process of submitting a resume to a specific recruiting web site is very time consuming. This is because the process does not only include the submission of a resume document but mostly requires the submission of every single date and detail of the own resume to specific fields. As to Maier et al. (2000) standard solutions would have a high impact here. If an applicant submits his/her resume in a format which is standardized (such as HR-XML or equivalent), the applicant will only once have to submit his/her resume and could send it to any company, attaching a company-specific application letter.

Another extension to this scenario would be the definition of a standard exchange format, which enables participation of many companies and applicants. If additionally a matchmaking web-service is integrating the applicants and vacant positions of many companies this network would enable an efficient and easy to use personnel recruiting service.

One suggestion is to use the GRAPPA matchmaking algorithms for this matching task within the HR domain. In the next section the HRNETAGENT project is described, which provides a solution to this idea.

6.2 Project Background

In 1999 the project HRNETAGENT has been launched by the Siemens AG CT together with Siemens Business Service and the Federal German Employment Agency[4]. The aim of the project was to define and implement a network which enables standardized submission of applications (candidates) and profiles of vacant postitions (centroids). A further idea was to include a configurable matchmaking service which provides a ranking of the applicants with respect to the positions. This process was meant to simplify the work of the HR departments and enable a more objective preselection of the relevant applicants for a position.

The Federal German Employment Agency stores the data of all persons who are reported currently being unemployed. The aim of the Federal German Employment Agency was to define a network which enables the integration of private employment agencies as well as companies into an extendable flexible and standardized network. This network – called HrNetAgent– is introduced in the next paragraphs.

6.3 HrNetAgent

The HrNetAgent is an application of Grappa for matching corporate job profiles with profiles of job applicants, stored in various data bases. The current version of HrNetAgent is a prototype system that has been developed for the Federal German Employment Agency, and demonstrates the feasibility of a partially automated approach to employment relaying. Based on its success, a full-fledged system is planned for the near future. The potential return on investment may be huge.

Figure 6.1 (see also Veit et al. (2002b), p. 866) shows the matchmaking architecture of the HrNetAgent system. A company specifies its job profiles to a designated GUI-agent, which takes the role of a requester agent in the system. The GUI-agent queries the matchmaker by sending to HrNetAgent the description of the open position which should be filled. Referring to the notion of this work (see Section 2.1.2), the scheme for specifying the open positions is the centroid. The back end of HrNetAgent consists of a collection of data sources wrapped by information agents and by a search controller that coordinates a number of search agents. For example one data source is the central database of the Federal German Employment Agency, in which all currently unemployed persons in Germany are stored. Others may be corporate skill databases; further databases can be easily integrated. Note that the database wrapper agents play the role of virtual provider agents in the software architecture.

The HrNetAgent human resources market is designed to find appropriate applicants from heterogeneous sources (e.g. the employee data of different companies). The results are continuously displayed in a homogenous way for

[4] http://www.arbeitsamt.de (01/01/2002)

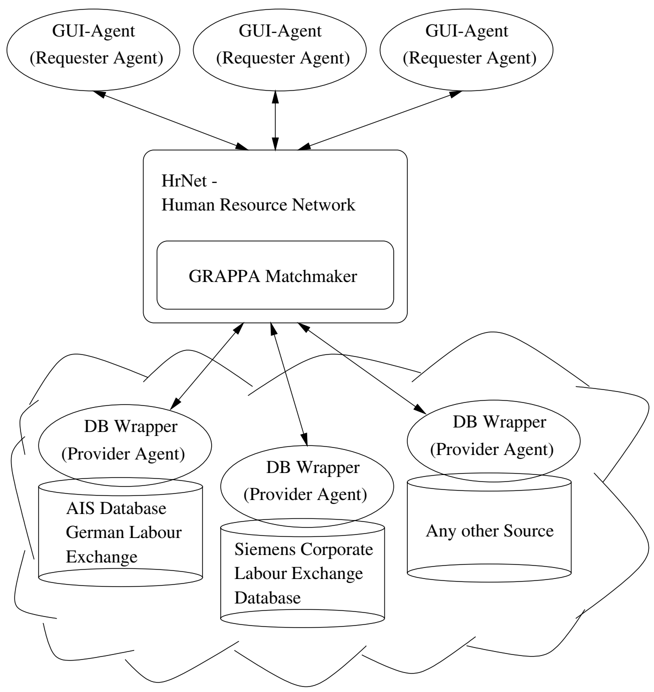


Figure 6.1. HrNetAgent System Overview

the user. These properties fulfill main points of the requirements for future job markets formulated by Maier et al. (2000).

Wrapper agents perform the task of query translation, connection handling, and result translation. They return a preselection of profiles to the matchmaker based on conditions extracted from the centroid profile. In HrNetAgent, the centroid and candidate schemes are converted to XML-DTDs which are considered as the document classes of these types. Matchmaking using the Grappa algorithms, thus, is based on a preselection of candidates. The most successful candidates for a job profile are stored in the local offer repository guaranteeing a fast access by the application. In addition, HrNetAgent offers an automated notification service via SMS, Fax, or Email.

6.3.1 Adaption of Grappa for HrNetAgent

As discussed in Chapter 4 and 5 the Grappa architecture is a highly flexible, adaptable, as well as a configurable framework for multidimensional matchmaking tasks in electronic negotiations.

Adoptions which are necessary to enable offer and request based matchmaking within electronic HR marketplaces are the definition of the XML-DTDs for the applicant (candidate), the job position (centroid) and the configuration – the mapping between candidate and centroid.

Due to the fact that all distance functions applied in the example are integrated into the GRAPPA implementation no more adoptions are needed to apply the GRAPPA matchmaker in a concrete matchmaking in this electronic market.

6.3.2 Results

The results the authors Maier et al. (2000) and Mülder (2000) draw is the lack of automized human resources management tools actual ERP systems. There exsits the need to automize personell recruitment, payroll accounting and project planning tasks.

To provide systems which enalble a partly automation in these domains it is necessary to support

- intelligent matchmaking and recruitment negotiation mechanisms,
- dynamic and scalable applications,
- standardized portal-based web interfaces for the users and
- flexible, easily adaptable data structures.

Several approaches are made in research and practice to provide these features. Some features are also included into ERP and HRIS systems which are currently applied in practice. Still there are major fields in which these systems can be enhanced. Here, especially the intelligence of matchmaking mechanisms is important. The user acceptance of partly automated recruitment systems does mainly correspond to the quality of the matches which is provided by the system.

6.4 Summary

In this chapter the real-world application of GRAPPA in the HRNETAGENT is introduced. Therefore, Human Resources Information Systems (HRIS) and their application within enterprise resource planning systems (ERP-systems) is shown in Section 6.1. Matchmaking approaches based on generic, market-based eNegotiation components are playing an increasingly important role within job exchanges on the Internet. Standards like HR-XML and other specific XML definitions enable the integration of job exchanges into ERP-systems and HRIS. Today's matchmaking components within job exchanges are mainly keyword based and do not support a decent quality of ranking.

In Section 6.3 the application of GRAPPA in the HR market is described. GRAPPA is featuring a job exchange which is on the one hand easy to be integrated into existing ERP-systems due to the fact that the underlying data structures are purely XML based. On the other hand it supports sophisticated matchmaking features which enable the integration of an intelligent matchmaking component.

In the next Chapter 7 the quality of the matchmaking results achieved by Grappa in the human resources domain is evaluated. The feasibility of enhanced matchmaking quality and the performance using the Grappa matchmaking framework is shown. Therefore, a comparison between the Grappa results and the results of a preselection by human resources experts is provided. Additionally, a comparison between the Grappa results and the matchmaking results achieved by the search engine Microsoft Index Server (TM) is provided which shows the matchmaking quality enhancement of multidimensional matchmaking with specific distance functions towards purely search-engine based matchmaking.

7 Empirical Evaluation

In this chapter the performance and validity of the GRAPPA framework is examined in six case studies. In the previous chapters a multidimensional matchmaking component for electronic negotiations was

- motivated,
- the general field and the research domains important for the approach were introduced,
- the related work in the domain of eNegotiations and matchmaking were illustrated,
- the framework was defined mathematically sound and complete,
- the implementation was described and outlined,
- as well as an application in a concrete market was introduced.

The aim of this chapter now is to show that, firstly, the GRAPPA approach which exists in a prototype implementation satisfies performance requirements to apply it in real-world applications (see Section 7.1). Secondly, but most importantly the goal is to show by six case studies that the matchmaking quality of the GRAPPA framework supports a quality which exceeds domain independent state-of-the-art technologies like the Microsoft Index Server (TM). The Microsoft Index Server (TM) is available as a commercial search engine available under the Windows NT/2000 Server operating system. This is shown by comparing both – the MS Index Server results as well as the GRAPPA matchmaking results – with the mean of the matchmaking results of a group of human resources specialists (human decision makers) from different domains (see Section 7.2).

7.1 Domain Independent Experiments with GRAPPA: Performance

As mentioned above this section shows the performance of the GRAPPA matchmaking library `de.siemens.zt.ik6matching` in the HRNETAGENT application. Table 5.1 in Section 5.3.2 shows a shortened version of the HRNETAGENT application configuration (MAP). The full configuration consists of ten dimensions which contain two to nine subdimensions. For each matchmaking (one candidate against one centroid) consequently 50 to 90 pairwise comparisons have to be calculated. Five of these comparisons in each matchmaking step are using the very time intensive FreeText distance function from information retrieval which has been introduced in Chapter 4.

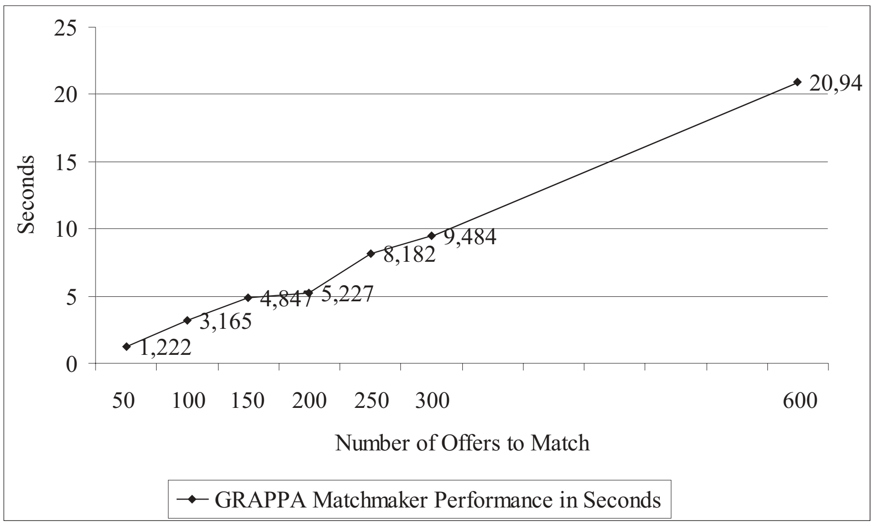


Figure 7.1. GRAPPA Performance in the HRNETAGENT Application

As expected the processing time of the GRAPPA matchmaker in dependence to the number of candidates which have to be matched against a centroid is growing linear $O(n)$ (see Figure 7.1). The results displayed here are achieved on a system which consists of a Dual Pentium III CPU with 900 MHz supporting 256MB of RAM. The GRAPPA matchmaker was installed separately – i.e. the RMI-server, the offer database as well as the front end components were running on a separate computer. A matchmaking centroid is sent to the EJB-matchmaker which was installed on the above mentioned system. The network latency with 0,125 seconds (on average) was subtracted from the measured matchmaking time (end-to-end).

In general the results are satisfying. The matchmaker takes about 20,94 seconds to match an amount of 600 candidates against a centroid. Taking into account that the EJB-matchmaker implementation is a prototype and was not optimized for computational time this result is acceptable. In practical application of the HRNETAGENT at most 300 candidates have to be matched against one centroid at the same time. This is due to the fact that a preselection using key features is initialized before the main matchmaking process. Consequently a user would have to wait 9,484 seconds for an ad-hoc matchmaking run.

In the next section the domain specific matchmaking quality of the Grappa framework is examined. Therefore several empirical evaluation runs with human experts were performed.

7.2 Domain Specific Experiments

Numerous approaches solving matchmaking problems exist and are being applied to various application domains. An important objective of the Grappa project was to create a generic framework that enables a quicker and better development of matchmaking solutions for different domains, such as human resources, procurement or logistics. An important question is how the quality of matchmaking can be determined for a specific domain. For instance, how good is a human resources matchmaking solution compared to a machine based one.

This question is in the focus of the following inquiries. In Section 7.2.1, a domain-independent approach to evaluating matchmaking algorithms is described based on a well-known IR model and applied to a concrete matchmaking application. In Section 7.2.2 the evaluation method is introduced. In Section 7.2.3 the measures for the evaluation of the quality of a matchmaking approach are presented. In Section 7.2.4 finally the results of a real life evaluation runs are presented, which was performed on the german human resources market.

7.2.1 Method for Case Studies

The relevance evaluation was done by taking human matchmaking results, generated by experts, as a reference solution. The matchmaking results computed by the Grappa matchmaker are correlated with the average of the results the five human experts performed. Additionally, a comparison with the results provided by the Microsoft Index Server (TM) is presented. The Microsoft Index Server (TM) also provides ratings that can be interpreted as percent values for relevance. The comparison shows that the Grappa matchmaker outperforms a matchmaker using a flat type structure without dimension or attribute clustering by far.

In Section 7.2.2 the settings for the case studies to perform the empirical evaluation of the Grappa framework are described. In Section 7.2.3 the evaluation measures are introduced and in Section 7.2.4 the results which have been recorded are presented. Finally in the conclusions, a statement about the results and an outlook is given.

7.2.2 Settings

The case studies for the Grappa framework were carried out in a specific application domain. An application of the Grappa matchmaker has to provide three different roles for participants: Candidates, centroids and the matchmaker agents. In this section as well as in Section 7.2.3 the domain independent evaluation method, which has been developed for the evaluation of the GRAPPA framework is described.

In the GRAPPA framework, the offering and requesting agents are both supporting multidimensional information objects, i.e. candidate and centroid descriptions which consist of many, semantically independent attributes.

These attributes are clustered into a multidimensional offer and request structure using multiple dimensions for clustering. A mapping between these structures is implemented using domain specific relevance functions to compare the data. Finally an over all score of each candidate for the current centroid is computed using the methodologies which are described in Chapter 4.

The setting of the case studies is defined as follows:

1. A group of three to five (if possible more) human experts from the application domain is chosen for an experiment.
2. One typical real-life example centroid is obtained from a concrete example which is, e.g. , supported by the human experts.
3. A set of 50 candidates is derived from different databases containing offer data. These databases are connected to the GRAPPA framework.
4. Each of the human experts obtains a hard copy of the 50 candidates and a hard copy of the centroid.
5. Each of the human experts generates a ranking of all 50 candidates regarding the concrete centroid. Consequently a relevance value between zero and 100 percent is specified for each candidate by each expert.
6. The GRAPPA matchmaker is applied to the same centroid and corresponding set of 50 candidates.
7. The experiment is then repeated for the same centroid and candidates, using Microsoft Index Server (TM) as a search engine. Therefore, the multidimensional offer information objects are extracted from the databases as pure ASCII text documents. The search query is generated from the request information object by also transforming it to a pure ASCII text document. Here, every information on structure as well as additional information such as meta-tags which are included into the offer and request objects are omitted.
8. The GRAPPA matchmaking results, the ranking results from the Microsoft Index Server, and the matchmaking results from the human experts are then normalized. This means that if, e.g. , the matchmaker ranked all 50 candidates between 10 and 80 percent, the corresponding interval $[0,1;\ 0,8]$ which contains all matchmaker results is mapped to the interval $[0;\ 1]$ under preservation of the order and the scope. The same normalization is performed on the human experts results and the Microsoft Index Server (TM) results. The normalization guarantees comparability.

9. From the opinions of the 5 human experts an arithmetic mean (average) is computed.
10. The results are displayed graphically as a correlation between the matchmaker and the Microsoft Index Server (TM) results with the human experts results as well as expressed in precision and recall values (defined in Section 7.2.3).

In the next section, the measures for the comparison of the Grappa approaches with the human experts opinions are explained.

7.2.3 Measures

In order to measure the results of the case studies, precision and recall are computed. Additionally, a graphical overview on the results is provided, as for example shown in Figures 7.4 and 7.6. Here, the results of the matchmaking by human experts are ranked in a descending order; the matchmaker results are inserted as a second graph. These figures show the correlation between the human resources experts' rankings and the matchmaker results as well as the Microsoft Index Server (TM) results, respectively.

Secondly, the results of the case studies are measured using the measures *precision* and *recall* from the information retrieval domain in a form adapted for the matchmaking evaluation. For this method the set of all candidates is divided into two sections: The relevant candidates and the non relevant candidates. Realistically the border is set to 60 percent, i.e. all candidates which reached a score of 60 percent or more are selected as relevant candidates and all below as non relevant. This value has proven to be a realistic value in the human resources domain. According to the human resources experts, on average 40 percent of the applicants are invited to an interview. Then, in the next step results of the matchmaker are considered. The Grappa matchmaker also returns the rankings for the complete set of candidates. Then, the ranked candidates are clustered into four categories:

1. **True positive:** Candidates which have been scored to a relevance of 60 percent or more by the matchmaker and which also have a score of 60 percent or more from the human experts.
2. **False positive:** Candidates which have been scored to a relevance of 60 percent or more by the matchmaker but have a score less than 60 percent from the human experts.
3. **False negative:** Candidates which did not reach a score of 60 percent by the matchmaker but reached a score over 60 percent by the human experts.
4. **True negative:** Contains all the rest.

The recall and precision measures from information retrieval which have been adapted for this evaluation are now defined as follows:

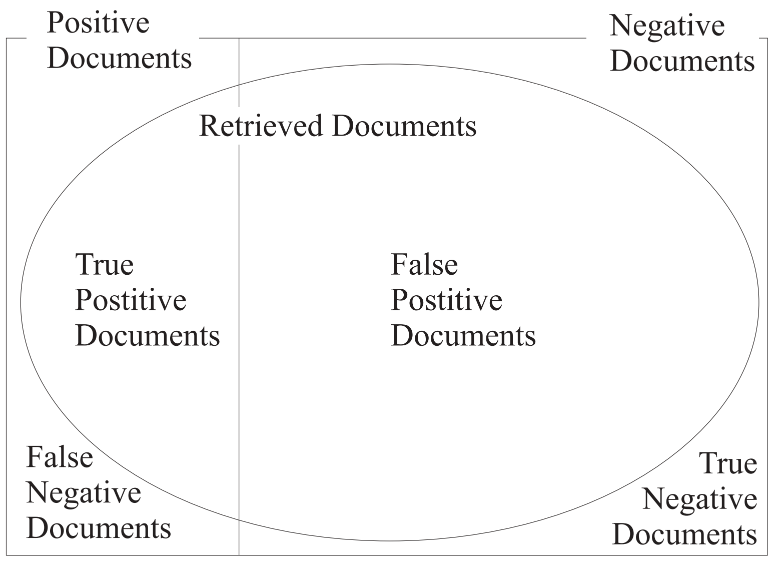


Figure 7.2. Precision and Recall

Definition 7.2.1 ***Precision, Recall***

See also Figure 7.2 (illustration see also Mädche 2002, p.194).

- **Precision:**

$$P = \frac{\#\text{true positive}}{\#\text{true positive} + \#\text{false positive}}$$

- **Recall:**

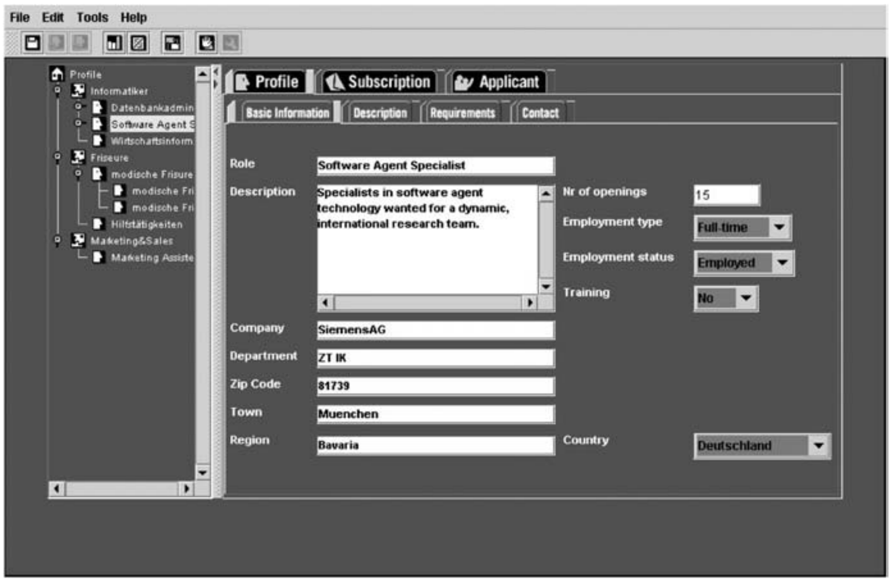
$$R = \frac{\#\text{true positive}}{\#\text{true positive} + \#\text{false negative}}$$

I.e., the *precision* reflects the number of the retrieved positive candidates compared to the number of all retrieved candidates; the *recall* reflects the number of retrieved positive candidates to the number of all positive candidates in the set.

In the next section these measures are applied to a specific evaluation with human resources experts from three different companies.

7.2.4 Results

The Grappa framework has been applied in a real-life application in the human resources domain. The matchmaking procedure has been instantiated to match unemployed persons as candidates against vacant positions as centroids. The prototype application which has been implemented using the Grappa matchmaking framework is the HrNetAgent. This application provides a GUI for submitting the centroid. It supports an agent based middleware, which includes wrapper agents for the data sources containing the candidate specifications as well as agents representing the centroids which are supported by the users. Figure 7.3 shows the GUI which enables the submission of centroids. As indicated on the left side of the frame, it is possible to define open positions as well as meta- and sub-positions. A meta-position would e.g. be a definition of the role "computer scientist" where a sub-position of this meta-position would e.g. be "software agent researcher".



Role	Software Agent Specialist		
Description	Specialists in software agent technology wanted for a dynamic, international research team.	Nr of openings	15
		Employment type	Full-time
		Employment status	Employed
		Training	No
Company	SiemensAG		
Department	ZT IK		
Zip Code	81739		
Town	Muenchen		
Region	Bavaria	Country	Deutschland

Figure 7.3. HrNetAgent: GUI for Centroid Specification

As indicated in Section 4.2.1 the centroids, candidates as well as the mapping between those are defined as instances of XML-DTDs. Table 5.1 in Section 5.3.2 shows an example for an instance of the mapping between candidate and centroid – here the mapping between the HrNetAgent candidate and centroid.

The generic evaluation methods, introduced in Sections 7.2.2 and 7.2.3 above are now applied on the HrNetAgent example. As human experts, several human resources specialists were matching sets of 50 candidates against one centroid. This procedure has been applied with three groups of human resources experts:

- Siemens AG Corporate Technology HR, Munich, Germany, see Section 7.2.4.
- Lufthansa Cargo AG HR, Frankfurt, Germany, see Section 7.2.4.
- University Hospital Giessen Internal Medicine HR, Giessen, Germany, see Section 7.2.4.

To provide a better reading the Figures 7.4 to 7.15 containing the evaluation results are provided in one row on pages 152 to 157.

Evaluation: Siemens AG Corporate Technology HR, Munich, Germany In the evaluation scenario with the Siemens AG Corporate Technology HR, a job description for a management assistant and a job description for a communication engineer is taken. In Figures 7.4 and 7.5 the results for the assistant are shown, in Figures 7.6 and 7.7 the results for the communication engineer. On the horizontal axis the number of the applicant is specified. The applicants are numbered along the quality, the average of the human resources department assigned to them, i.e. the applicant number 1 was rated highest by the HR experts, whereas the right-most applicant (number 50) reached the lowest score.

The graphical representation of the results is designed as follows: For each set of applicants two figures are integrated. The first (e.g. Figure 7.4) shows the correlation between Grappa matchmaker results as well as the average of the human resources experts. The second figure (e.g. Figure 7.5) shows the correlation between the Microsoft Index Server (TM) results as well as the same average of the human resources experts as in the first figure.

In the management assistant example, the Grappa matchmaker reached precision (p) and recall (r)

$$p_{\text{ma}}^{\textsc{Grappa}} = 0,60 \qquad r_{\text{ma}}^{\textsc{Grappa}} = 0,69$$

where the Microsoft Index Server reached only

$$p_{\text{ma}}^{\text{MS IS}} = 0,50 \qquad r_{\text{ma}}^{\text{MS IS}} = 0,08.$$

In the communication engineer example, the Grappa matchmaker reached precision (p) and recall (r)

$$p_{\text{ce}}^{\textsc{Grappa}} = 0,43 \qquad r_{\text{ce}}^{\textsc{Grappa}} = 0,75$$

where the Microsoft Index Server reached only

$$p_{\text{ce}}^{\text{MS IS}} = 0,29 \qquad r_{\text{ce}}^{\text{MS IS}} = 0,50.$$

Evaluation: Lufthansa Cargo AG HR, Frankfurt, Germany In the system architect example, the GRAPPA matchmaker reached precision (p) and recall (r)

$$p_{\text{sa}}^{\text{GRAPPA}} = 0,70 \qquad r_{\text{sa}}^{\text{GRAPPA}} = 0,78$$

(see also Figure 7.8 and 7.9) where the Microsoft Index Server reached only

$$p_{\text{sa}}^{\text{MS IS}} = 0,60 \qquad r_{\text{sa}}^{\text{MS IS}} = 0,17.$$

In the procurement referee example, the GRAPPA matchmaker reached precision (p) and recall (r)

$$p_{\text{pr}}^{\text{GRAPPA}} = 0,71 \qquad r_{\text{pr}}^{\text{GRAPPA}} = 0,63$$

(see also Figure 7.10 and 7.11) where the Microsoft Index Server reached only

$$p_{\text{pr}}^{\text{MS IS}} = 0,0 \qquad r_{\text{pr}}^{\text{MS IS}} = 0,0.$$

Evaluation: University Hospital Giessen Internal Medicine HR, Giessen, Germany In the medical doctor example, the GRAPPA matchmaker reached precision (p) and recall (r)

$$p_{\text{md}}^{\text{GRAPPA}} = 0,16 \qquad r_{\text{md}}^{\text{GRAPPA}} = 1,00$$

(see also Figure 7.12 and 7.13) where the Microsoft Index Server reached worse recall but better precision

$$p_{\text{md}}^{\text{MS IS}} = 1,00 \qquad r_{\text{md}}^{\text{MS IS}} = 0,67.$$

In the laboratory assistant example, the GRAPPA matchmaker reached precision (p) and recall (r)

$$p_{\text{mta}}^{\text{GRAPPA}} = 0,71 \qquad r_{\text{mta}}^{\text{GRAPPA}} = 0,63$$

(see also Figure 7.14 and 7.15) where the Microsoft Index Server reached only

$$p_{\text{mta}}^{\text{MS IS}} = 0,17 \qquad r_{\text{mta}}^{\text{MS IS}} = 0,63.$$

7.2.5 Discussion

Values of precision and recall that are close to 100 percent mean *good*, values tending to zero mean *poor* compared to human experts judgement. As it has been shown in many machine learning approaches in the past (compare Salton 1989), algorithmic matching and information retrieval optimizes recall but reaches only moderate precision values. Human experts mostly reach good precision values but fail in optimizing recall. For this reason it is most promising to combine highly sophisticated machine matchmaking like GRAPPA with human experts.

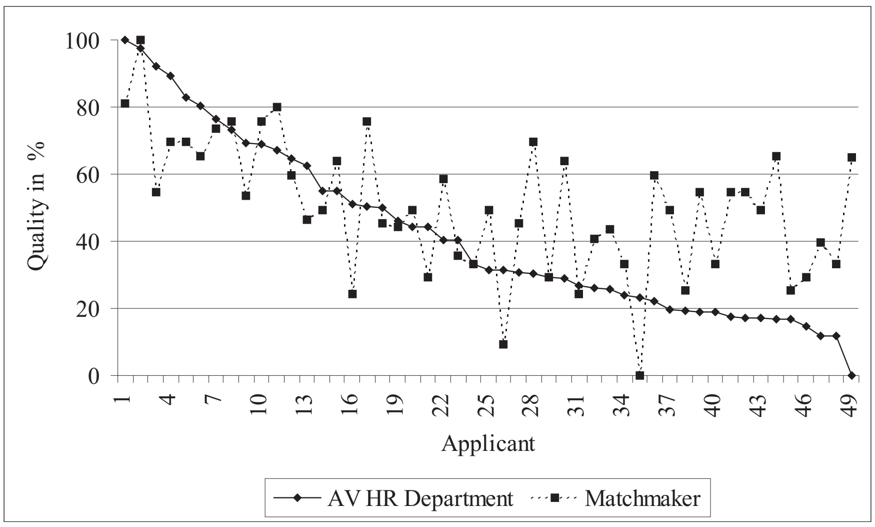


Figure 7.4. Evaluation of Applicant Data for *Assistants at Siemens AG*: AV HR Department vs. Grappa Matchmaker

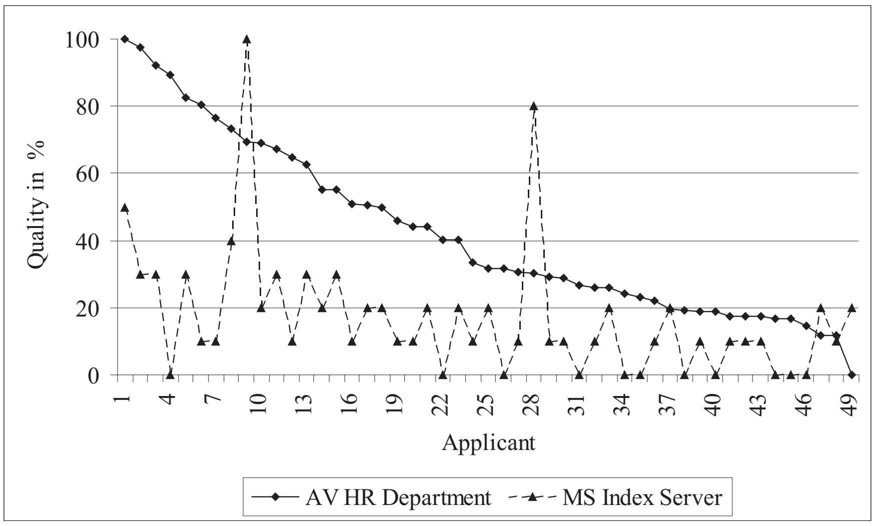


Figure 7.5. Evaluation of Applicant Data for *Assistants at Siemens AG*: AV HR Department vs. MS Index Server (TM)

Discussion of Results The best procedure here is to firstly match all possible candidates against a given centroid using machine matching profiting from the high recall. Then human experts can use the results of the machine

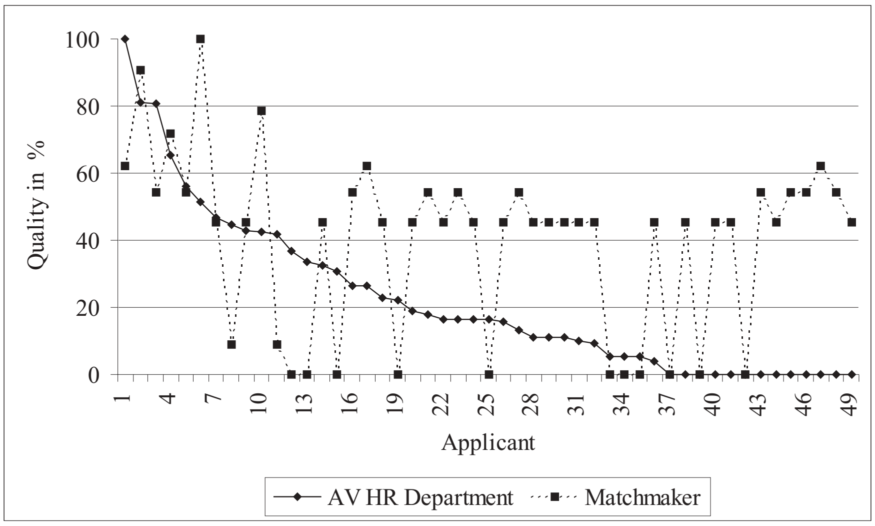


Figure 7.6. Evaluation of Applicant Data for *Communication Engineers at Siemens AG*: AV HR Department vs. GRAPPA Matchmaker

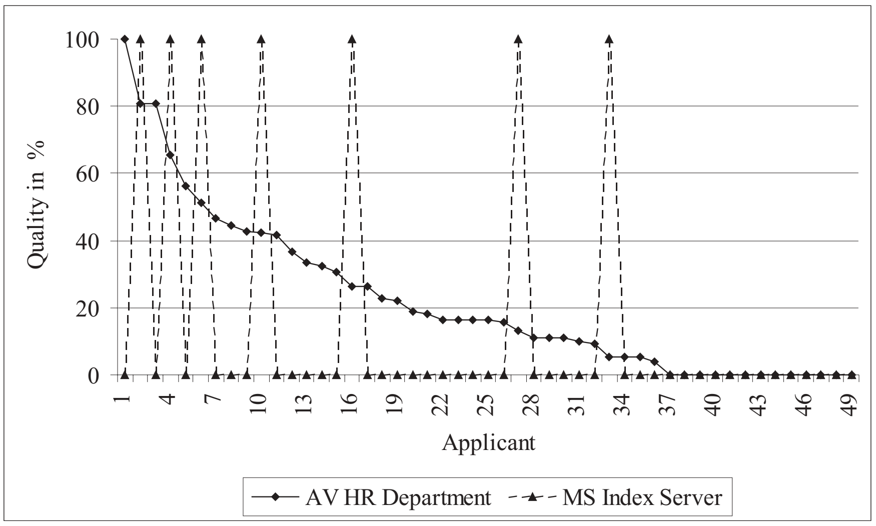


Figure 7.7. Evaluation of Applicant Data for *Communication Engineers at Siemens AG*: AV HR Department vs. MS Index Server (TM)

matchmaking and optimize precision. Similar observations were also made by Mädche 2002, p.219 in the automated ontology learning domain.

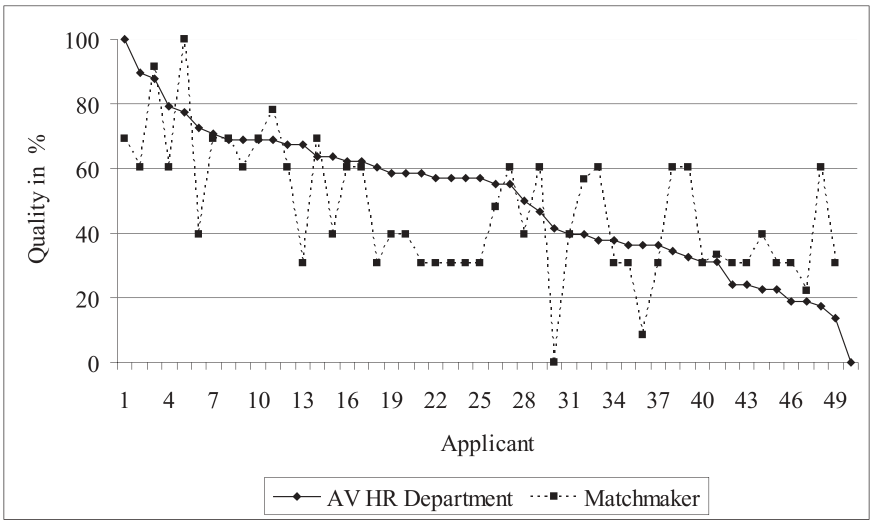


Figure 7.8. Evaluation of Applicant Data for *System Architects at Lufthansa Cargo AG*: AV HR Department vs. Grappa Matchmaker

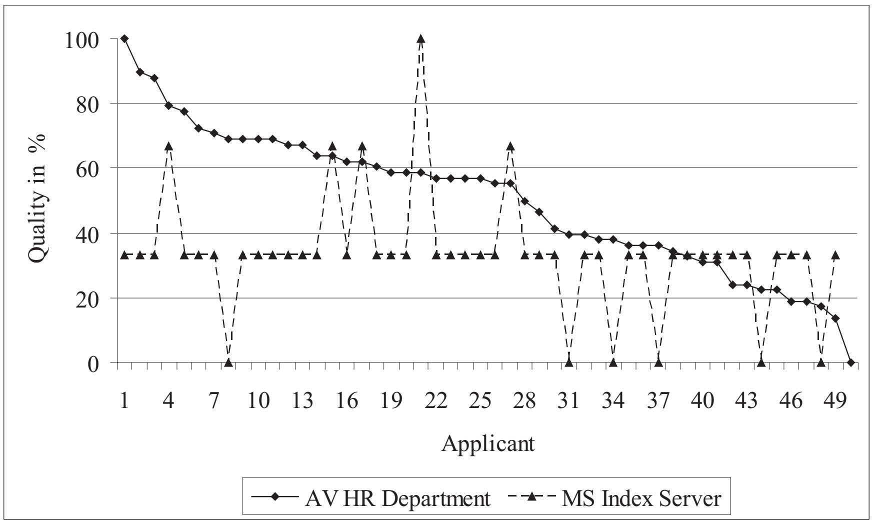


Figure 7.9. Evaluation of Applicant Data for *System Architects at Lufthansa Cargo AG*: AV HR Department vs. MS Index Server (TM)

A good example for this is Figure 7.4. In particular, Grappa provides fairly high ratings for two candidates that were ranked very poorly by the human experts (candidates number 44 and 50). They are examples of cases where a human expert easily sees that they are poor matches, but where an automated

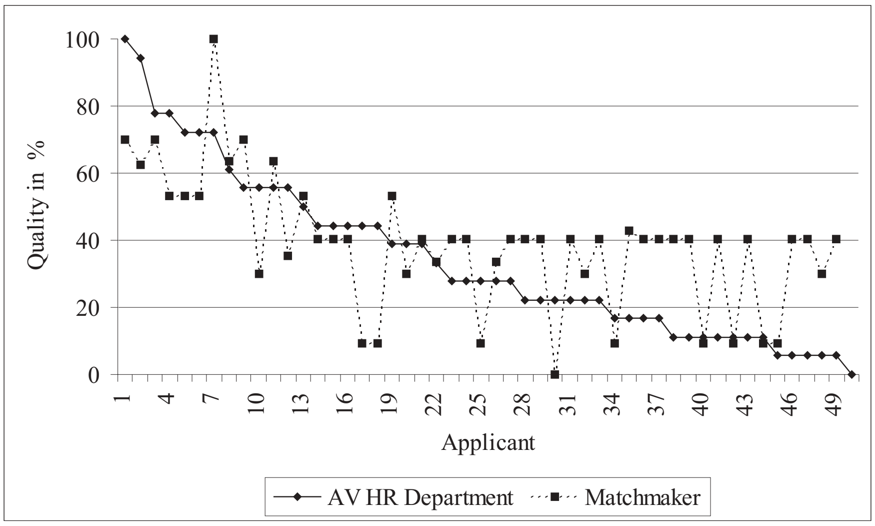


Figure 7.10. Evaluation of Applicant Data for *Procurement Referees at Lufthansa Cargo AG*: AV HR Department vs. GRAPPA Matchmaker

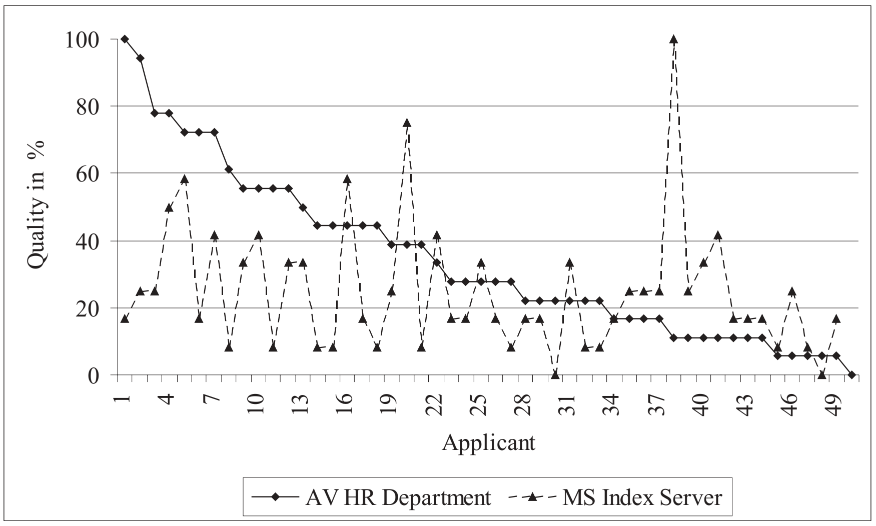


Figure 7.11. Evaluation of Applicant Data for *Procurement Referees at Lufthansa Cargo AG*: AV HR Department vs. MS Index Server (TM)

tool like GRAPPA may encounter problems. On the other hand, GRAPPA provides good ratings for most applicants that were ranked highly by the human experts, leading to a good recall value. If Figure 7.4 is compared with Figure 7.5 it is easy to see that the matchmaking capability of the GRAP-

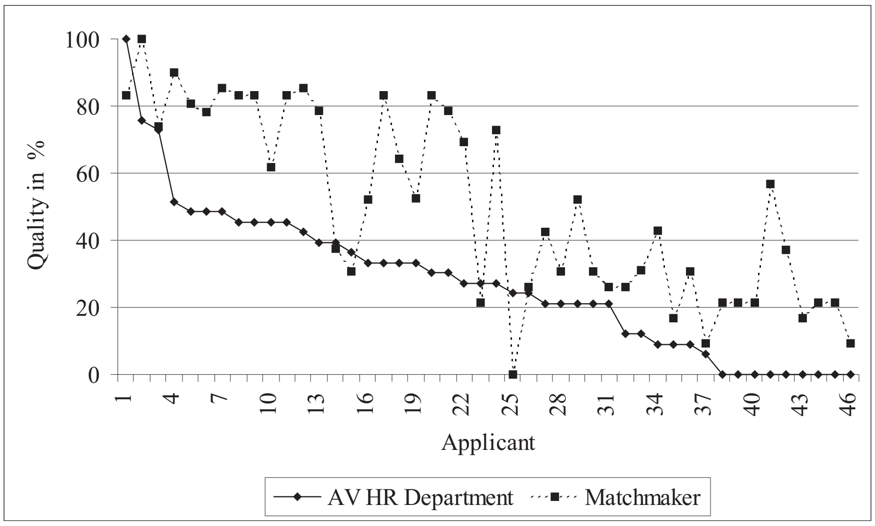


Figure 7.12. Evaluation of Applicant Data for *Medical Doctors at University Hospital Giessen*: AV HR Department vs. Grappa Matchmaker

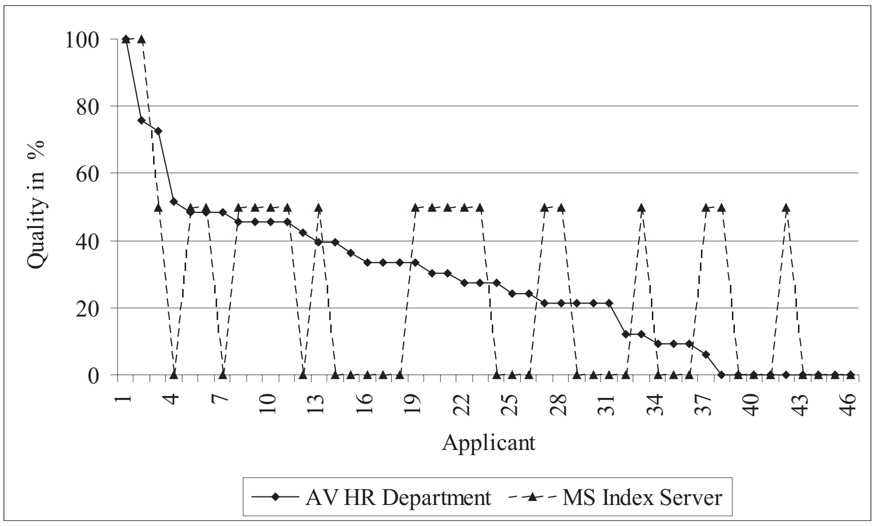


Figure 7.13. Evaluation of Applicant Data for *Medical Doctors at University Hospital Giessen*: AV HR Department vs. MS Index Server (TM)

pa matchmaker outperforms the capabilities of the Microsoft Index Server (TM) by far. Only in the case of the *medical doctor* scenario, the Microsoft Index Server outperformed the Grappa matchmaker in precision. Here the multidimensional document structure of Grappa worked inversely: Relevant

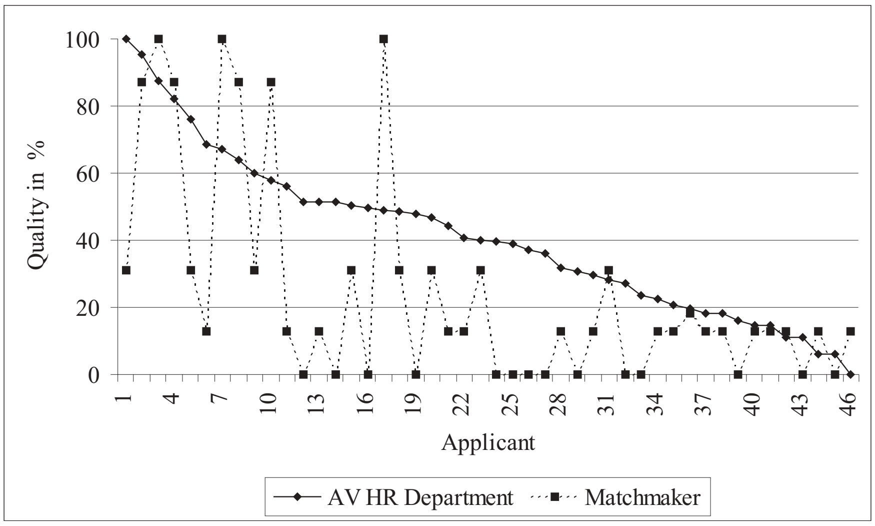


Figure 7.14. Evaluation of Applicant Data for *Laboratory Assistants at University Hospital Giessen*: AV HR Department vs. Grappa Matchmaker

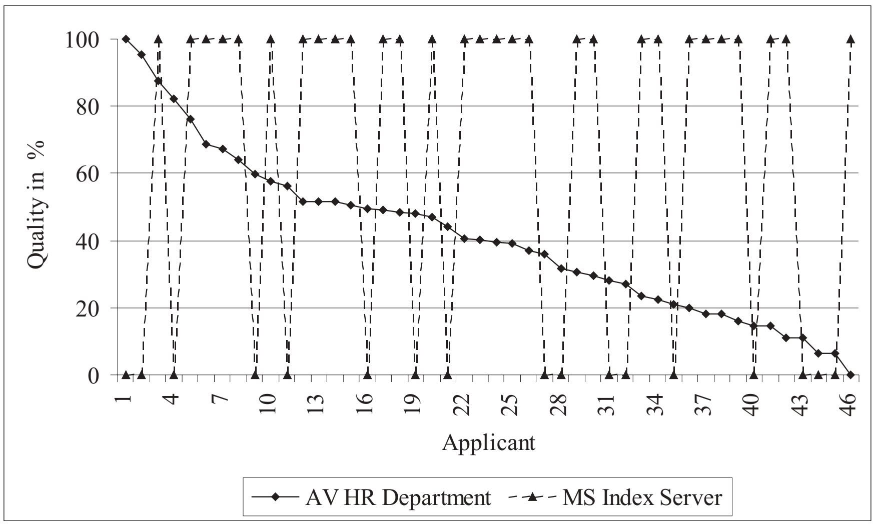


Figure 7.15. Evaluation of Applicant Data for *Laboratory Assistants at University Hospital Giessen*: AV HR Department vs. MS Index Server (TM)

information was not mapped towards each other in the candidate and centroid structure. This example shows that a domain specific customization of the Grappa matchmaker would provide even better matchmaking results.

In some of the evaluation cases (see e.g. Figure 7.6 and Figure 7.7) the matchmaker finds only few criteria to compute a relevace among the candidates. This can bee seen by eight relevance layers of the Grappa matchmaker and only two relevance layers of the Microsoft Index Server (TM), respectively.

The worst performance of the Grappa matchmaker in comparison to the average of the human experts opinion can be seen in Figure 7.14 and Figure 7.15. Here the Grappa matchmaker does not find the best candidate (candidate number 1). Still, even in this case, the Grappa matchmaker outperforms the Index Server by far.

Generally said the Grappa matchmaker optimizes the recall, but not precision. Especially in a domain like HR, where wrong decision are costly, a system that candidates a high recall and a somewhat lower precision is preferrable to a system that is precise but fails to find some potentially good candidates.

Limitations of the Approach Of course, the current experiment is based on a specific configuration of the matchmaker. Hence, based on the results of the evaluation the matchmaking logic will be improved to reach better recall and precision values. As discussed in Section 4.2.3, it shows that by incorporating user feedback and using information about the user's behavior valuable input to the analysis of the quality of matchmaking can be obtained. One additional important aspect that the case studies with the human experts in the human resources domain showed was that their ratings are highly subjective. There exist considerable differences across experts opinions as regards how different criteria should be weighted. To this effect, it is obvious that the possibility of Grappa to provide personalized matchmaking, i.e., create one matchmaker configuration per user, is a key feature of this approach.

A central problem to the evaluation method in general is the difficulty to determine a "standard"-expert opinion. As it can be easily seen in Figure 7.16 the correlation among several HR experts ranking the same set of data is often not closely correlated. This particular example is taken from the management assistant example in the Siemens use case. The general direction in ranking of several HR experts is the same. But in concrete examples several experts may vary strongly. Taking into account that the experts opinion is normally distributed it is formally correct to take the arithmetic mean of the experts opinions available. The major tradeoff which is encountered here is the fact that only a small set of HR experts can be considered. But, at least, it is better to consider a small set than only single experts.

It should be noted that the correlation between the Grappa matchmaker's results and the HR experts mean is of the same or better quality then the correlation among the individual HR experts opinions.

In Figure 7.17 the results from the MS Index Server from Figure 7.5 are shown after revision. In the revision the two runaways (matching result 9

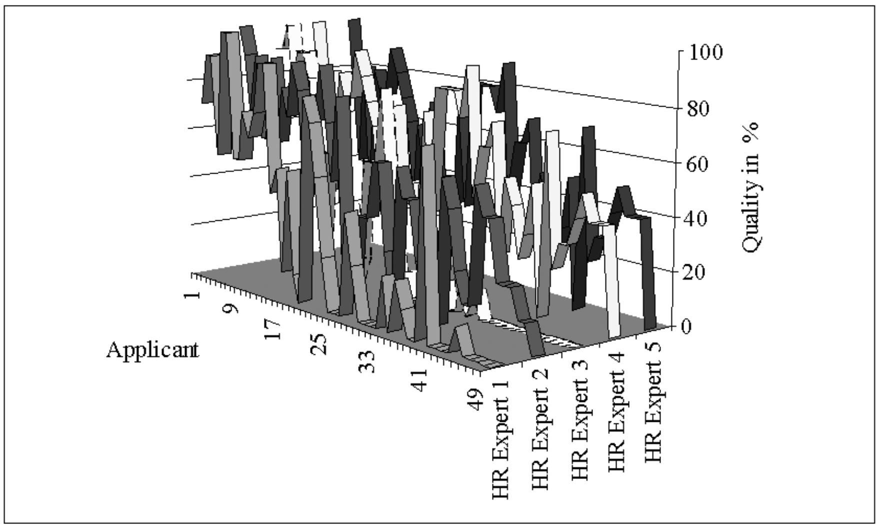


Figure 7.16. Correlation of five HR Experts Ranking the same Set of Data

and matching result 28) have been removed. Afterwards the MS Index Server results are a newly mapped to the interval [0; 1].

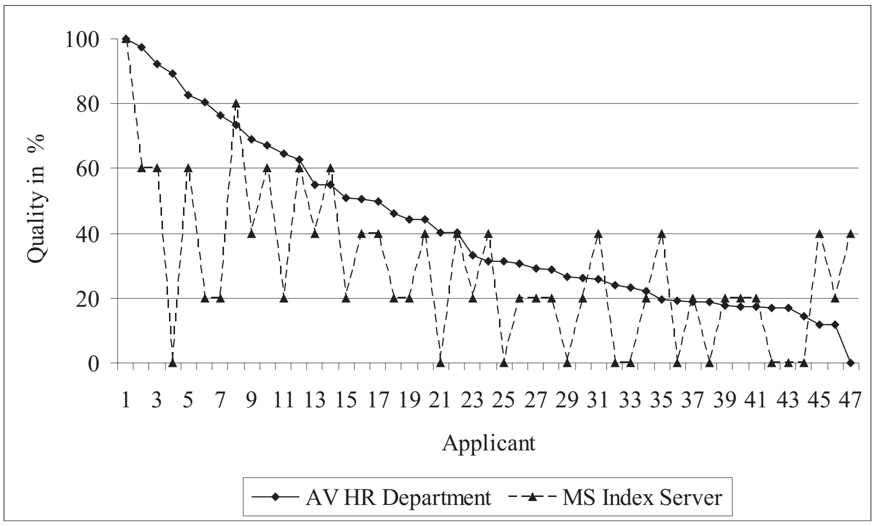


Figure 7.17. Evaluation of Applicant Data for *Assistants at Siemens AG*: AV HR Department vs. MS Index Server (TM), Revised

After the application of this procedure, the MS Index Server reaches

$$p_{\mathrm{ma}}^{\mathrm{MS\ IS}} = 1,00 \qquad r_{\mathrm{ma}}^{\mathrm{MS\ IS}} = 0,17.$$

Thereby the MS Index Server does not reach the values of the GRAPPA matchmaker in the recall dimension but exceeds the GRAPPA results in the precision.

Finally, the described limitations show, that a large number of case studies have to be performed to prove the quality of a new matchmaking method. Especially the runnaway elimination makes traditional approaches again more attractive.

7.3 Summary

In this chapter the performance and the matchmaking quality of the GRAPPA approach are discussed. The performance of the GRAPPA matchmaker is satisfying although the prototypical implementation is not runtime optimized. This is discussed in Section 7.1.

The more interesting and central point of this chapter is the empirical evaluation of the matchmaking results. The precision and recall values which are reached by the GRAPPA matchmaker are over all reasonable in comparison to automated, non-structured matchmaking and ranking tools such as the Microsoft Index Server (TM) (see Section 7.2). Of course a lot has to be done in the future. Intelligent domain specific matchmaking components will become more and more important in electronic markets. Therefore, new distance functions have to be defined and implemented, since these are the crucial functions which determine the applicability of matchmaking in a larger number of real-world applications. A promising way to provide sophisticated matchmaking frameworks is to provide sets of distance functions for classes of problems and to allocate these distance function sets to the classes. This approach will provide expandability and may integrate also machine learning into distance function definitions.

In the next section this work is reviewed and concluded. Additionally an outlook is provided which contains interesting perspectives for future research, in which the author is involved currently and which he will address in future.

8 Conclusions and Outlook

Beyond the e-Business hype, the development of electronic markets has strongly gained importance within the last years. This tendency gave spirit to the research area of electronic markets within economics and computer science. Many mechanisms from these research fields have been integrated into electronic market platforms lately.

The motivation for this work was taken from the urgent need for powerful, generic and adaptable matchmaking mechanisms in the domain of electronic markets. Today's offer and request based systems begin to enable highly dynamic and application dependent mechanisms of negotiating electronic goods and contracts. Hence, mechanisms are needed which capture the decision capability of human experts in ranking offers and requests. In this work's approach decision capability is integrated into distance functions which partly mirror human experts opinions.

Defining such matchmaking mechanisms is hard. This work contributes in some fields to the definition of electronic market mechanisms. Several approaches which have been proposed within the research fields in the last years are outlined, analyzed and enhanced. The genericity of the approach provided in this work exceeds the known approaches and provides a flexible, dynamic and – most importantly – domain independent matchmaking framework.

Central aspects and contributions of the framework defined in this work are:

- Request and offer profiles in electronic markets are often complex (e.g. matchmaking job profiles against applicants) and require multidimensional matchmaking. Often, a combination of existing methods is adequate to deal with different aspects of matchmaking.
- Request and offer information is distributed and heterogeneous. Thus, distributed search and ontology mapping may be required to achieve comparable profiles.
- Request and offer profiles differ depending on the application domain; also the underlying business logic to determine the quality of a match (distance functions) is very domain-specific in part. The framework supports this variety.
- The framework restricts the effort of developing new electronic market solutions by enabling reuse of existing profiles and business logic. In par-

ticular, developing a matchmaking solution using Grappa requires only coding efford, and is done mainly through customization of XML-files.
- The matchmaking framework assists market developers by supporting a clear process for building matchmaking solutions; the enforcement of this process is supported by appropriate tools.
- Participating software agents in the matchmaking framework cannot succeed in a vacuum, but need to comply with industrial standards. Therefore, beyond relying on agent standards such as FIPA, which can be achieved by using FIPA-compliant agent platforms, it is necessary to serve the integration needs and capabilities of today's e-business platforms, which are in this framework based on Enterprise Java Beans (EJB) application servers.

8.1 Review of This Work

In this work a solution for the problem of multidimensional matchmaking among offers and requests in different types of electronic markets is presented. The solution to the problem which is formalized and implemented in the Grappa framework addresses two central problems:

- The definition of a generic multidimensional matchmaking framework with a mapping among heterogeneous offer and request profiles.
- The definition of domain independent and domain dependent distance functions for the computation of both, atomic and complex distances among offer and request instances.

This work captures the complete process of

- *conception,*
- *definition,*
- *modelling,*
- *implementation,*
- *application* and
- *evaluation*

of the multidimensional matchmaking problem in electronic negotiations.

Chapter 1 introduces this work and provides the motivation of this research. An overview on this work as well as a project history is provided. Additionally the results of this work which have partially been published within the last years are referenced.

Chapter 2 provides the definitions of the central terms and concepts which are treated throughout this work. This chapter captures the view in the research domain and prepares the notion multidimensional multidimensional matchmaking for the entire work. Here, the *conception* of the multidimensional matchmaking problem is considered.

It might be mentioned that the definition of the notion "multidimensional" differs somewhat from the understanding of other researchers in the domain. There are several different understandings of this term within the community which are in detail concerned in Chapter 3.

Chapter 3 is focussed on the contributions in the field which seemed, from the authors point of view, relevant to this work. Special attention is spent to the approaches where mechanisms from traditional computer science are applied to economic problems. There are three groups who especially influenced the definition of the GRAPPA framework. Within the economic domain the contributions of Weinhardt and Gomber (1999) and coworkers with the AMTRAS approach as well as Bichler (2000a) and coworkers with the contributions concerning multi-attribute negotiations provided strong influences. From the computer science domain it is Sycara (1999) and coworkers with their LARKS matchmaking approach and their classification of information agents as well as Subrahmanian et al. (2000) with their IMPACT approach.

Chapter 4 finishes the *conception* of the multidimensional matchmaking framework. In this chapter the *definition* as well as the *modelling* of the framework are described. Firstly a non-formal framework overview is presented. This overview is then executed in a formal definition of the GRAPPA Matchmaking Base (GMB) as well as the GRAPPA Matchmaking System (GMS) which is developed throughout the first part of the chapter. Here, also the definitions and considerations about the distance functions applied in the GRAPPA approach are outlined. With its definition, provided in this chapter, the GRAPPA framework exceeds the existing approaches in

- genericity – the flexibility to define arbitrary offer and request structures.
- applicability – to the authors knowledge it is the first matchmaking approach providing a generic and application independent mapping for heterogeneous offers and requests.
- scalability – due to the open and transparent definition of the offer and request types the framework can be expanded or shrinked to the application demand.
- parallelism – the parallel matching in several dimensions and attributes exceeds the known approaches.

In Chapter 5 the *implementation* of the generic matchmaking framework GRAPPA is introduced. The focus of the first part of the chapter to provide an overview on the applied state-of-the-art technologies (JAVA, RMI-Server, EJB, MS SQL-Server, etc.). In the second part, the GRAPPA package which is composed to the JAVA package `de.siemens.zt.ik6matching` is provided. Here the most important interfaces and functions are outlined. An excerpt of the UML class diagram of the framework is also provided.

Chapter 6 provides the *application* of the matchmaking framework in a concrete domain. For that the Human Resources (HR) domain and the problem of matching vacant positions (requests) against job seekers (offers) was chosen. Because of its complexity, the HR domain is capable of showing the feasibility of matchmaking using GRAPPA. The application of GRAPPA is prepared by the introduction of the most important notions from the state-of-the-art information system in the HR-domain in the first part of the chapter.

Finally, Chapter 7 provides the *evaluation* of the GRAPPA matchmaking performance and quality. The most important results of the GRAPPA matchmaker are achieved in the quality evaluation. Here, the GRAPPA matchmaking quality is compared with the matchmaking quality of a state-of-the-art matchmaker, the Microsoft Index Server (TM). To show the real-world applicability both – the results of the GRAPPA matchmaker as well as the results of the MS Index Server (TM) – are compared with the opinion of human resources experts. Human experts from three domains were interviewed and tested: Five personnel managers from Siemens CT AG Munich, Germany, three experts from Lufthansa Cargo AG, Frankfurt a.M., Germany as well as three assistant medical directors from the University Hospital Giessen, Germany. The results of this evaluation show that the GRAPPA matchmaker exceeds the state-of-the-art matchmaking quality in all tested cases.

The matchmaking performance is tested by incrementally increasing the number of matches to be executed. The performance of the GRAPPA matchmaker shows that it is – although implemented as prototype which is not performance optimized – applicable in medium size business domains. Underlying better performing application servers as well as hardware, business level can be reached. This is shown in current industrial tests done at the project site at Siemens CT[1].

As a concluding and comprehending chapter, this chapter finishes the work.

8.2 Future Work

The research described in this work shows the immense need of transferring effective, scalable and tested algorithms from the computer science domain to economics. The contributions made in this work help showing that there is a great potential of optimizing business applications using results from the computer science domains of information retrieval and Multi-Agent Systems.

8.2.1 Limitations of This Approach

The GRAPPA approach has been designed to be a generic matchmaking framework. This has been shown in the application of the methods within

[1] Siemens Corporate Technology, Competence Center for Intelligent Autonomous Systems, Munich, Germany. Currently finishing the business version of the `de.siemens.zt.ik6matching` library.

the human resources domain. However, of course specific distance functions have been integrated for the relevance computation within human resources. Therefore, it would be of high interest to apply the Grappa architecture within other domains and to integrate distance functions from other domains in order to obtain further matchmaking quality results.

This approach does not integrate ontologies and learning capabilites. In both ways the architecture can be extended. Such extensions promise to improve the matchmaking quality regarding precision and recall.

Finally it would be helpful to provide graphical-user-interface based tools to enable an easy development of matchmaking applications. The current version of the software library enables the definition of matchmaking applci-ations without much coding work. Still the system designer has to implement the offer and request structures in XML files. The automation of this process would enable an easy application of matchmaking e.g. within web services.

8.2.2 Extension of Multidimensional Matchmaking Using Ontologies and Learning

An important aspect is the impact of domain dependent ontologies for the Grappa approach. Currently intense research is done in this field. Mädche et al. (2002) provide significant results in ontology learning for the semantic web. Adding learning ontologies to matchmaking solutions will multiply their effect in business applications, especially if domain dependent ontologies are underlied.

One approach here can be to integrate an ontology into the distance functions for free text. Here, especially the domain dependence of concrete ontologies plays a crucial role. Using this feature, special distance functions for different application domains and within those the specific free text elements can be obtained and exploited.

Finally, it will be of great interest to integrate reinforcement learning strategies to agents participating in matchmaking applications. Using reinforcement learning, an agent attempts to learn how to select an action within the environment to maximize the total win recieved within the environment (see Dzeroski 2002). Some promising approaches from this domain are currently investigated for the Grappa framework at the Siemens CT Research labs in Munich (see also Fiehn and Mueller 2003).

8.2.3 Further Application of Multidimensional Matchmaking in the Finance Domain

It will be challenging to apply the Grappa framework to matchmaking problems in other application domains such as commodity exchanges or trading in financial markets. There, the potential of the applicability and the scalability of the Grappa approach can be tested and matchmaking results within another discipline can be evaluated.

Matchmaking is one important component in electronic markets. Providing a high market performance, developing appropriate business-models for application domains as well many other problems are challenging research fields within market engineering.

Important work which is to be done in near future is the integration of matchmaking and domain dependent distance functions into generic electronic market research such as e.g. the eFIT[2] project (see Neumann et al. 2002). From the integration of such projects new, domain independent results can be obtained to improve the matchmaking algorithms.

As also the authors Paolucci et al. (2000) show with their A-Match matchmaker within the Warren system that portfolio management using intelligent matchmaking components is an up to date research area. Here, matchmaking among risk profiles of customers and risk specifications of financial instruments play a central role. It is an ongoing research question, how portfolio management can be partially automized using matchmaking mechanisms.

8.2.4 Multidimensional Matchmaking, Web Services, and the Grid

The domain of web services is a new and ongoing field in practice and research. A web service is a "...programmable component that provides a service and is accessible over the Internet. Web services can be standalone or linked together to provide an online functionality"[3]. Web services are used to integrate business services and make them accessible via the Internet. Several actual technologies are used to enable web services. Enterprise Java Beans, Microsoft's .NET (TM) platforms are the most prominent technologies which are applied to provide web services. Also, web services are a central enabling technology for grid computing scenarios. Within such a scenario Grappa can be applied as a matchmaker for offers and requests in grid markets.

Current research projects within the DAML (The DARPA Agent Markup Language)[4] – the DAML-S (DAML for Services) – provide mechanisms for web service definition. E.g. the ATLAS (Agent Transaction Language for Advertising Services)[5] uses agents to provide web services. The authors Paolucci et al. (2002) show how a semantic matching between advertisments and requests based on the profile section of a DAML-S description can be performed.

[2] http://www.e-fit.org (06/25/2002)
[3] see http://www.embedded.com/story/OEG20020125S0103 (08/10/2002) E-Cyclopedia
[4] http://www.daml.org (11/02/2002)
[5] http://www.daml.ri.cmu.edu (10/20/2002)

It will be a challenge to provide the Grappa framework which is available as an EJB component, for web services and grid applications. Here two aspects will primarily be considered. Firstly, to provide Grappa as a web service itself and secondly, to provide a Grappa based domain dependent matchmaking facility to match web service capabilites of offering and requesting web services in order to find the best matching counterparts.

A web service based application of Grappa promises to enable the usage of multidiemsional matchmaking in a wide variety of business applications.

References

Arisha, K., T. Eiter, S. Kraus, F. Ozcan, R. Ross, and V.S. Subrahmanian (1999, March/April). Impact: A platform for collaborating agents. *IEEE Intelligent Systems 14*(2), 64–72.

Bauer, B., J. P. Müller, and J. Odell (2001). Agent UML: A formalism for specifying multiagent software systems. *International Journal of Software Engineering and Knowledge Engineering 11*(3), 207–230.

Beam, C., M. Bichler, R. Krishnan, and A. Segev (1999). On negotiations and dealmaking in electronic markets. *Information Systems Frontiers 1*(1-2), 241–258.

Beam, C. and A. Segev (1997). Automated negotiations: A survey of the state of the art. *Wirtschaftsinformatik 39*(3), 263–268.

Beam, C. and A. Segev (1998). Auctions on the internet: a field study. Technical report, Fisher Center for Management and Information Technology, University of California, Berkeley, 98-WP-1032.

Bellifemine, F., A. Poggi, and G. Rimassa (2001). Developing multi agent systems with a fipa-compliant agent framework. *Software - Practice And Experience 31*, 103–128.

Benyoucef, M., H. Alj, M. Vézeau, and R. K. Keller (2001, July). Combined negotiations in e-commerce: Concepts and architecture. *Electronic Commerce Research Journal - Special issue on Theory and Application of Electronic Market Design 1*(3), 277–299.

Benyoucef, M., A. Hakim, and R. Keller (2001, September). An infrastructure for rule-driven negotiating software agents. In *Proceedings of the Twelfth International Workshop on Database and Expert Systems Applications (DEXA 2001), IEEE*, Munich, Germany, pp. 737–741.

Benyoucef, M. and R. Keller (2000). An evaluation of formalisms for negotiations in e-commerce. In *DCW*, pp. 45–54.

Benyoucef, M., R. Keller, S. Lamouroux, J. Robert, and V. Trussart (2000, February). Towards a generic e-negotiation platform. In *Proceedings of the Sixth Conference on Re-Technologies for Information Systems*, Zurich, Switzerland, pp. 95–109.

Bichler, M. (2000a). An experimental analysis of multi-attribute auctions. *Decision Support Systems 29*(3), 249–268.

Bichler, M. (2000b). Trading financial derivatives on the web - an approach towards automating negotiations on OTC markets. *Information Systems Frontiers 1*(4), 401–414. Kluwer Academic Publishers.

Bichler, M. (2001). *The Future of E-Markets – Multidimensional Market Mechanisms.* Cambridge, U.K.: Cambridge University Press.

Bichler, M., C. Beam, and A. Segev (1998a). An electronic broker for business-to-business electronic commerce on the internet. *International Journal of Cooperative Information Systems 7*, 315–331.

Bichler, M., C. Beam, and A. Segev (1998b). OFFER: A broker-centered object framework for electronic requisitioning. In *Trends in Distributed Systems for Electronic Commerce*, pp. 154–165.

Bichler, M. and J. Kalagnanam (2002, June). Winner determination and bidding languages in multi-attribute auctions. Technical Report RC22478 (W0206-018).

Bichler, M., M. Kaukal, and A. Segev (1999). Multi-attribute auctions for electronic procurement. In *Proceedings of the 1st IBM IAC Workshop on Internet Based Negotiation Technologies*, Yorktown Heights, USA, pp. 154–165.

Bichler, M., S. Strecker, and G. Kersten (2002). Engigeering of negotiations. *Group Decision and Negotiation* (Special Issue). forthcoming.

Bichler, M. and H. Werthner (2000, 4-8 September). Classification framework for multidimensional, multi-unit procurement auctions. In *Proceedings of the DEXA Workshop on Negotiations in Electronic Markets*, Greenwich, U.K., pp. 1003–1009.

Budimir, M. and C. Holtmann (2001). *e-Finance: Innovative Problemlösungen für Informationssysteme in der Finanzwirtschaft*, Chapter The Design of Innovative Securities Markets: The Case of Asymmetric Information, pp. 175–196. Springer.

Budimir, M., C. Holtmann, and D. Neumann (2002). Design of a best execution market. *Revue Bancaire et Financière/Bank en Financiewezen 66*(2-3), 138–146.

Calisti, M. and B. Faltings (2001). Agent-based negotiations for multi-provider interactions. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS).* forthcoming.

Caple, J. and M. Altarace (2001). The art of ejb-deployment. JavaWorld.

Chaudhury, A., H. Rao, and S. Rathnam (1991, November). What can computer programs do to facilitate negotiation processes? In *Proceedings of the Ninth International Conference on Information Systems Development*, Atlanta, USA, pp. 269–284.

Decker, K., K. Sycara, and M. Williamson (1996, December). Matchmaking and brokering. In *International Conference on Multi-Agent Systems (ICMAS96).*

Decker, K., K. Sycara, and M. Williamson (1997, August). Middle-agents for the internet. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, pp. 578–583.

Durfee, E. and J. Rosenschein (1994). Distributive problem solving and multi-agent systems: Comparison and examples. In *Proceedings of the 13th International Workshop on Artificial Intelligence*, pp. 94–104.

Dzeroski, S. (2002). Relational reinforcement learning for agents in worlds with objects. In *Proceedings of the AISB'02 Symposium on Adaptive Agents and Multi-Agent Systems*, pp. 1–8.

Eiter, T. and H. Mannila (1997). Distance measures for point sets and their computation. *Acta Informatica 34*, 109–133.

Eiter, T., D. Veit, J. P. Müller, and M. Schneider (2001, September). Matchmaking for structured objects. In *Proceedings of the Third International Confer-*

ence on Data Warehousing and Knowledge Discovery (DaWaK 2001), pp. 186–194.

Faratin, P. (2000, December). *Automated Service Negotiation Between Autonomous Computational Agents.* Dissertation, Department of Electronic Engineering, Queen Mary & Westfield College, London, Great Britain.

Faratin, P., C. Sierra, and N. Jennings (2000). Using similarity criteria to make negotiation trade-offs. In *Proceedings of the 4th International Conference on Multi-Agent Systems (ICMAS 2000)*, Boston, USA, pp. 119–126.

Faratin, P., C. Sierra, N. Jennings, and P. Buckle (1999, July). Designing responsive and deliberative automated negotiators. In *AAAI Workshop on Negotiation: Settling Conflicts and Identifying Opportunities*, Orlando Florida, USA, pp. 12–18.

Fiehn, B. and J. P. Mueller (2003). User-adaptive matchmaking. In *Proceedings of the Workshop on Agent Mediated Electronic Commerce V (AMEC-V), held at the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS).*

Finin, T., R. Fritzson, D. McKay, and R. McEntire (1994). KQML as an agent communication language. In *Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94)*, pp. 456–463.

FIPA Members (1997). Fipa 97 specification, part 1, agent management. Technical report, Foundation for Intelligent Physical Agents (FIPA).

Genesereth, M. and R. Fikes (1992, June). Knowledge interchange format, version 3.0 reference manual. Technical Report Logic-92-1, Computer Science Department, Stanford University. `http://www-ksl.stanford.edu/knowledge-sharing/papers/kif.ps` (01/01/2002).

Guttman, R. and P. Maes (1998). Cooperative vs. competitive multi-agent negotiations in retail electronic commerce. In *Cooperative Information Agents*, pp. 135–147.

Guttman, R. and P. Maes (1999). Agent-mediated integrative negotiation for retail electronic commerce. *Lecture Notes in Computer Science 1571*, 70–90.

Guttman, R., P. Maes, A. Chavez, and D. Dreilinger (1997, May). Results from a multi-agent electronic marketplace experiment. In *Poster Proceedings of Modeling Autonomous Agents in a Multi-Agent World (MAAMAW'97)*, Ronneby, Sweden.

Guttman, R., A. Moukas, and P. Maes (1998a). Agent-mediated electronic commerce: A survey. *Knowledge Engineering Review 13*(2), 147–159.

Guttman, R., A. Moukas, and P. Maes (1998b). Agents as mediators in electronic commerce. *EM-Electronic Markets 8*(1).

Harned, E. (2001). A Java RMI server framework. IBM Developer Network.

Hoffner, Y., C. Facciorusso, S. Field, and A. Schade (1999). *Distributed Applications and Interoperable Systems*, Chapter Distribution issues in the design of and implementation of a virtual market place, pp. 405–422. Kluwer Academic Publishers. ISBN 0-7923-8527-6.

Jennings, N. R., P. Faratin, A. R. Lomuscio, S. Parsons, C. Sierra, and M. Wooldridge (2001). Automated negotiation: prospects, methods and challenges. *International Journal of Group Decision and Negotiation 10*(2), 199–215.

Jennings, N. R., K. Sycara, and M. Wooldridge (1998). A roadmap of agent research and development. *International Journal Autonomous Agents and Multi-Agent Systems 1*, 7–38.

Jha, S., P. Chalasani, O. Shehory, and K. Sycara (1998, May). A formal treatment of distributed matchmaking. In *In Proceedings of the Second International Conference on Autonomous Agents (Agents 98)*, pp. 457–458.

Kalin, M. (2001). *Object-Oriented Programming in Java*. DePaul University - Prentice Hall. ISBN 0130198595.

Kersten, G. (1999). *Decision Support Systems for Sustainable Development - A Resource Book of Methods and Applications*, Chapter Decision making and decision support, pp. 29–51. Norwell, IDRC and Kluwer Academic Publishers. ISBN 0-7923-8527-6.

Kersten, G. and D. Cray (1996). Perspectives on representation and analysis of negotiations: Towards cognitve support systems. *Group Decision and Negotiation 5*(4-6), 433–469.

Kersten, G., S. Noronha, and J. Teich (2000). Are all e-commerce negotiations auctions? In *Proceedings of the Fourth International Conference on the Design of Cooperative Systems (Coop'2000)*, Sophia-Antipolis, France.

Klemperer, P. (1999). Auction theory: A guide to the literature. *Journal of Economic Surveys 13*(3), 227–286.

Koppius, O. (1998). Electronic multidimensional auctions: Trading mechanisms and applications. In V. Homburg, M. Janssen, and M. Wolters (Eds.), *Edispuut Workshop'98: Electronic Commerce — Crossing Boundaries*, Rotterdam, The Netherlands.

Koppius, O., M. Kumar, and E. van Heck (2000, July). Electronic multidimensional auctions and the role of information feedback. In H. R. Hansen, M. Bichler, and H. Mahrer (Eds.), *Proceedings of the 8th European Conference on Information Systems (ECIS'2000)*, Volume 1, Vienna, Austria, pp. 461–468. Association for Information Systems.

Kumar, M. and S. Feldman (1998, August). Internet auctions. In *Proceedings of the 3rd USENIX Workshop on Electronic Commerce*, Boston, USA, pp. 49–60.

Kuokka, D. (1990). *The Deliberative Integration of Planning, Execution, and Learning*. Ph. D. thesis, School of Computer Science, Carnegie Mellon University.

Kuokka, D. and L. Harada (1995, June). On using KQML for matchmaking. In *Proceedings of the first international Conference on Multi-Agent Systems*, pp. 239–245. AAAI Press.

Kuokka, D. and L. Harada (1996). Integrating information via matchmaking. *Journal of Intelligent Information Systems 6*(2-3), 261–279.

Lipsey, R. (1989). *An Introduction to Positive Economics* (7 ed.). Weidenfeld and Nicholson, London.

Lo, G. and G. Kersten (1999). Negotiation in electronic commerce: Integrating negotiation support and software agent technologies. In *Proceedings of the 29th Atlantic Schools of Business Conference*. CD-ROM Proceedings.

Lomuscio, A., M. Wooldridge, and N. R. Jennings (2001, March). *Agent-Mediated Electronic Commerce: A European AgentLink Perspective*, Volume 1991, Chapter A Classification Scheme for Negotiation in Elctronic Commerce. Springer-Verlag Lecture Notes in AI.

Lux, A. and D. Steiner (1995). Understanding cooperation: an agent's perspective. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pp. 261–268. AAAI Press.

Madanmohan, T., G. Kersten, S. Noronha, M. Kersten, and D. Cray (2000). *Decision Support Systems for Sustainable Development*, Chapter Learning Negotiations with Web-based Systems: The Case of IIMB. Boston, USA: IDRC and Kluwer Academic Publisher.

Mädche, A. (2002). *Ontology Learning for the Semantic Web*, Volume 665 of *Series in Engineering and Computer Science*. Kluwer Academic Publishers.

Mädche, A., S. Staab, N. Stojanovic, R. Studer, and Y. Sure (2002). *Spinning the Semantic Web*. MIT Press. forthcoming.

Maier, M. and M. Gollitscher (2001). Überlegungen zum Skill-Matching-Modul eines Leitstands für den regionalen, zwischenbetrieblichen Personalaustausch. Technical Report FWN-2001-007. (in German).

Maier, M., K. Kronewald, and P. Mertens (2000). Vernetzte Jobbörsen und Unternehmensnetzwerke – eine Vision. *WIRTSCHAFTSINFORMATIK Sonderheft IT und Personal 42*, 124–131. (in German).

Morris, J. and P. Maes (2000, April). Negotiating beyond the bid price. In *Workshop Proceedings of the Conference on Human Factors in Computing Systems (CHI 2000)*, The Hague, The Netherlands.

Mülder, W. (2000). Personalinformationssysteme - Entwicklungsstand, Funktionalität und Trends. *Wirtschaftsinformatik Sonderheft IT und Personal 42*, 98–106. (in German).

Müller, J. P. (1997). *The design of intelligent agents. Lecture Notes of Articial Intelligence*, Volume 1077. Springer-Verlag.

Neumann, D., C. Holtmann, H. Weltzien, C. Lattemann, and C. Weinhardt (2002, October). Towards a generic e-market design. In *Proceedings of the Second IFIP Conference "e-Commerce, e-Business and e-Government", (I3E'2002)*, Lissabon, Portugal, pp. 289–303.

Nodine, M., W. Bohrer, and A. H. H. Ngu (1999, August). Semantic brokering over dynamic heterogenous data sources in InfoSleuth. In *Proceedings of the Fifteenth International Conference on Data Engeneering*, pp. 358–365.

Oliver, J. (1996a). A machine learning approach to automated negotiation and prospects for electronic commerce. *Journal of Management Information Systems 13(3)*, 83–112.

Oliver, J. (1996b). On artificial agents for negotiation in electronic commerce. In *Proceedings of the 29th Annual Hawaii International Conference on Systems Sciences*, Hawaii, USA, pp. 337–346.

Oliver, J. (1996c). *On Artificial Agents for Negotiation in Electronic Commerce*. Ph. D. thesis.

Oliver, J. (1997). Artificial agents learn policies for multi-issue negotiation. *International Journal of Electronic Commerce 1(4) (Summer)*, 49–88.

Paolucci, M., T. Kawamura, T. Payne, and K. Sycara (2002). Semantic matching of web services capabilities. In *Proceedings of the First International Semantic Web Conference (ISWC)*. forthcoming.

Paolucci, M., Z. Niu, K. Sycara, C. Domashnev, S. Owens, and M. V. Velsen (2000). Matchmaking to support intelligent agents for portfolio management. In *In Proceedings of AAAI2000.* (demosession).

Poslad, S., P. Buckle, and R. Hadingham (2000). The FIPA-OS agent platform: Open source for open standards. In *Proceedings of the 5th International Converence and Exhibition on the Practical Application of Intelligent and Multiagent Technology (PAAM).*

Raiffa, H. (1996). Lectures on negotiation analysis. *Program on Negotiation, Cabridge, MA: Cambridge Unviersity Press.*

Salton, G. (1988). Automatic text indexing using complex identifiers. In *Proceedings of the ACM SIGIR'88 Conference*, pp. 135–143.

Salton, G. (1989). *Automatic Text Processing.* Addison-Wesley.

Salton, G., J. Allan, and C. Buckley (1993). Approaches to passage retrieval in full text information systems. In *Proceedings of the ACM SIGIR'93 Conference*, pp. 49–58.

Sandholm, T. (1999). Automated negotiation. *Communications of the ACM 43 (3)*, 84–85.

Sandholm, T. (2000). eMediator : a next generation electronic commerce server. In *International Conference on Autonomous Agents (Agents 2000)*, pp. 341–348.

Schmid, B. (1993). Elektronische Märkte. *Wirtschaftsinformatik 35*(2), 465–480. in German.

Schmid, B. (1998). *Management – Handbuch Electronic Commerce. Grundlagen, Strategien, Praxisbeispiele*, Chapter Elektronische Märkte - Merkmale, Organisation und Potentiale. Munich: Vahlen. (in German).

Schoop, M. (2002). Electronic markets for architects - the architecture of electronic markets. *Information Systems Frontiers 4*(3), 285–302.

Sierra, C., P. Faratin, and N. R. Jennings (1999). Deliberative automated negotiators using fuzzy similarities. In *Proc. EUSFLAT-ESTYLF Joint Conference on Fuzzy Logic*, Palma de Mallorca, Spain, pp. 155–158.

Staab, S. (2002). Wissensmanagement mit Ontologien und Metadaten. *Informatik Spektrum 25*(3), 194–209. (in German).

Strecker, S. and C. Weinhardt (2001, April 25-27 2001). Wholesale electricity trading in the deregulated german electricity market. In *2001: An Energy Odyssey?*, Houston, TX. International Association for Energy Economics (IAEE).

Ströbel, M. (2000a, July). Effects of electronic markets on negotiation processes. In H. R. Hansen, M. Bichler, and H. Mahrer (Eds.), *ECIS 2000 – A Cyberspace Odyssey. Proceedings of the 8th European Conference on Information Systems (ECIS 2000)*, Volume 1, Vienna, Austria, pp. 445–452. Association for Information Systems: Vienna University of Economics and Business Administration.

Ströbel, M. (2000b, September). A framework for electronic negotiations based on adjusted-winner mediation. In K. Bauknecht, S. Kumar Madria, and G. Pernul (Eds.), *Electronic Commerce and Web Technologies: First International Conference, EC-Web 2000, London, U.K., September 4-6 2000, Proceedings*, Volume 1875 of *Lecture Notes in Computer Science*, Berlin et. al., pp. 1020–1028. Database and Expert Systems Applications (DEXA): Springer.

Ströbel, M. (2001a, May). Communication design for electronic negotiations on the basis of xml schema. In *Proceedings of the tenth international World Wide Web conference on World Wide Web*, Hong Kong, China, pp. 9–20.

Ströbel, M. (2001b). Design of roles and protocols for electronic negotiations. *Electronic Commerce Research Journal, Special Issue on Market Design 1(3)*, 335–353.

Ströbel, M. and M. Stolze (2001). A matchmaking component for the discovery of agreement and negotiation spaces in electronic markets. In *Proceedings of the Group Decision and Negotiation Conference, La Rochelle France*, pp. 61–75.

Ströbel, M. and C. Weinhardt (2002). The montreal taxonomy for enegotiations. *Group Decision and Negotiation* (Special Issue). forthcoming.

Subrahmanian, V. S., P. Bonatti, J. Dix, T. Eiter, S. Kraus, F. Ozcan, and R. Ross (2000, June). *Heterogenous Agent Systems.* MIT Press. ISBN: 0262194368.

Sycara, K. (1999). In-context information management through adaptive collaboration of intelligent agents. In M. Klusch (Ed.), *Intelligent Information Agents: Cooperative, Rational and Adaptive Information Gathering on the Internet.* Springer Verlag.

Sycara, K., J. Lu, and M. Klusch (1998). Interoperability among heterogenous software agents on the internet. Technical Report CMU-RI-TR-98-22, The Robotics Institute Carnegie Mellon University, Pittsburgh.

Sycara, K., J. Lu, M. Klusch, and S. Widoff (1999). Dynamic service matchmaking among agents in open information environments. *ACM SIGMOD Record 28 (1), Special Issue on Semantic Interoperability in Global Information Systems*, 47–53.

Sycara, K., S. Widoff, M. Klusch, and J. Lu (2002). Larks: Dynamic matchmaking among heterogeneous software agents in cyberspace. *Autonomous Agents and Multi-Agent Systems 5*, 173203.

Teich, J., H. Wallenius, and J. Wallenius (1999). Multiple-issue auction and market algorithms for the world wide web. *Decision Support Systems 26*, 49–66.

Teich, J., H. Wallenius, J. Wallenius, and A. Zaitsev (1999). A multiple unit auction algorithm: Some theory and a web implementation. *Electronic Markets 9*(3), 199–205.

Ueyama, J., E. Roberto, and M. Madeira (2001, March). An automated negotiation model for electronic commerce. In *Proceedings of The Fifth International Symposium on Autonomous Decentralized Systems*, Dallas, Texas, USA, pp. 29–37.

Varian, H. (1995). Economic mechanism design for computerized agents. In *Proceedings of the First Usenix Workshop on Electronic Commerce*, pp. 13–22.

Veit, D., J. P. Müller, M. Schneider, and B. Fiehn (2001). Matchmaking for autonomous agents in electronic marketplaces. In *Proceedings of the Fifth International Conference on Autonomous Agents*, Montreal, Canada, pp. 65–66.

Veit, D., J. P. Müller, and C. Weinhardt (2002a, July). An empirical evaluation of multidimensional matchmaking. In *Proceedings of the Workshop on Agent Mediated Electronic Commerce IV (AMEC-IV), held at the First Inter-*

national Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS).

Veit, D., J. P. Müller, and C. Weinhardt (2002b). Multidimensional matchmaking for electronic markets. *Applied Artificial Intelligence 16*(9-10), 833–869.

Vorhees, E. (1994). Query expansion using lexical-semantic relations. In *Proceedings of the ACM SIGIR'94 Conference*, pp. 61–69.

Weinhardt, C. (2002). *Taschenbuch der Wirtschaftsinformatik und Wirtschaftsmathematik* (second ed.)., Chapter Internet-Auktionen. Verlag Harri Deutsch. (in German, forthcoming).

Weinhardt, C. and P. Gomber (1999, January). Agent mediated off exchange trading. In *Proceedings of the 32nd Hawaii Conference on System Sciences (HICSS).*

Weinstein, P. and W. Birmingham (1997). Service classification in a proto-organic society of agent. In *Proceedings of the IJCAI-97 Workshop on Artificial Intelligence in Digital Libraries.*

Weiss, G. (1999). *Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence.* MIT Press.

Wiederhold, G. (1993). Intelligent Integration of Information. In *Proceedings of ACM SIGMOD Conference on Management of Data*, Washington, DC, pp. 434–437.

Wooldridge, M. and N. Jennings (1995). *Intelligent Agents - Theory, Architectures and Languages.* Springer Verlag, Lecture Notes in Aritificial Intelligence 890. ECAI Workshop.

Wooldridge, M., J. P. Müller, and M. Tambe (1996). *Intelligent Agents II.* Springer Verlag, Lecture Notes in Aritificial Intelligence 1037. IJCAI Workshop.

Wurman, P. (1999). *Market Structure and Multidimensional Auction Design for Computational Economies.* Ph. D. thesis, Computer Science and Engineering, University of Michigan.

Wurman, P., M. Wellman, and W. Walsh (1998, May). The michigan internet auctionbot: A configurable auction server for human and software agents. In *Proceedings of the second International Conference on Autonomous Agents (Agents-98)*, Minneapolis, USA, pp. 301–308.

Wurman, P., M. Wellman, and W. Walsh (2001). A parameterization of the auction design space. *Games and Economic Behavior 35*, 304–338.

Zaniolo, C. (1991). The logical data language (ldl): An integrated approach to logic and databases. Technical Report STP-LD-328-91, MCC.

Zlotkin, G. and J. Rosenschein (1996). Mechanisms for automated negotiation in state oriented domains. *Journal of Artificial Intelligence Research 5*, 163–238.

Index